QL Today

Volume 4
Issue 3
Sept./October
1999

ISSN 1432-5454

The Magazine about QL, QDOS, Sinclair Computers, SMSQ...

Demonstrated at the Paris QL Show

The Colour Drivers for the Q40 exist!

Your Q40 display could soon look like this:

Stella

Arnould Nazarian explains the concepts

Contents

3 Editorial
4 News
7 Stella — *Arnould Nazarian*
15 SeekQL 2.09 — *Al Feng*
18 Gee Graphics! (on the QL?) - part 12 — *Herb Schaaf*
25 Adventures on the QL Part 4 - Voyage of the Beano — *Darren D. Branagh*
28 CAUTION: HOTKEYS - Don't Burn Your Fingers - Part 1 — *David Denham*
34 You and Your Software - Just good Friends? - Part 4 - Problems not solutions — *Geoff Wicks*
35 The Paris International QL Show — *Wolfgang Lenerz*
36 QLTdis - part two — *Norman Dunbar*
42 Programmer's choice: using IF and SELect - 1 — *Mark Knight*
44 SCART Connections - Part 1 — *Richard Cooke*
46 EPSON Printer Control Codes — *Ian Pizer*
46 More THINGS... - Part 2 — *Jochen Merz*
48 Q-Help - D'you Need Somebody? — *Roy Wood*
52 QL-PC Transfers — *Geoff Wicks*
52 SUQCESS - A database-manager — *Wolfgang Uhlig*
54 Byts of Wood — *Roy Wood*

Advertisers

in alphabetical order

Jochen Merz Software 17, 27
Just Words! 34
PROGS - Van der Auwera 29
QBOX USA 45
QBranch 10, 11
Quanta 6
RWAP QL Software 33
TF Services 21

QL *Today*

ISSN 1432-5454

German office & Publisher:

Jochen Merz Software	Tel. +49 203 502011
Im stillen Winkel 12	Fax +49 203 502012
47169 Duisburg	Box1 +49 203 502013
Germany	Box2 +49 203 502014
	email: JMerz@j-m-s.com

English office:

Q Branch	Tel. +44 1903 200212
The Bank Volt	Mobile +44 836 745501
6 Coronation Buildings	Fax +44 1903 208070
Ham Road	Fax +44 1273 381577
Sussex BN11 2NN	email: qbranch@qbranch.demon.co.uk
United Kingdom	

Editor:

Dilwyn Jones	Tel. +44 1248 354023
41 Bro Emrys	Fax +44 1248 354023
Tal-Y-Bont, Bangor	email: dilwyn.jones@dj.softnet.co.uk
Gwynedd	
United Kingdom LL57 3YT	

QL *Today* is published bi-monthly, our volume begins on 15th of May. Subscriptions begin with the current issue at the time of sign up. Please contact the German or English office for current subscription rates.
Payment in DM (drawn on a German bank) can be made by either Cheque or Euro-Cheque, Payment in £ (drawn on an English bank) can be made by Cheque. Cheques should be made payable to Jochen Merz Software (German office) or QBranch (English office).

Credit Card holders may subscribe by either calling or sending their Credit Card number and Expiry date. Credit cards will be charged in DM (German office) or in £ (English office).

We welcome your comments, suggestions and articles. YOU make **QL *Today*** possible. We are constantly changing and adjusting to meet your needs and requirements. Articles for publication should be on a 3.5" disk (DD or HD) or sent via Email or into one of the JMS-BBS's. We prefer ASCII, Quill or text87 format. Pictures may be in _SCR format, we can also handle GIF or TIF To enhance your article you may wish to include Saved Screen dumps. PLEASE send a hardcopy of all screens to be included. Don't forget to specify where in the text you would like the screen placed.

Article and Advertising deadlines are as follows:

Issue 1: 30 April	Issue 2: 30 June
Issue 3: 30 August	Issue 4: 30 October
Issue 5: 30 December	Issue 6: 30 February

Editorial

Dilwyn Jones

The Paris show recently and other less publicised events have given the QL scene a much needed boost with some long awaited good news.
Tony Tebby demonstrated the much fabled "colour drivers" in action on a Q40 there to prove (as if it needed proving) that they are not vapourware. Once fully implemented on the Q40, the QXL and Aurora versions may follow. Marcel Kilgus is known to harbour wishes to port "colour drivers" to QPC in due course, and Jochen Merz is known to want to get a system for the Milan computer too.

Another Tony Tebby technology, the Stella system, is explained by one of its keenest proponents, Arnould Nazarian, in this issue. It has been discussed in the QL Users Mailing List for example, but I thought it might be nice to give this subject a wider stage in this issue. Jonathan Dent was also at the Paris show from Switzerland and confirmed that work is well advanced on the TCP/IP system for QL systems. Much of the 'low-level' work is done, but there is no email client or Web browser for it yet.

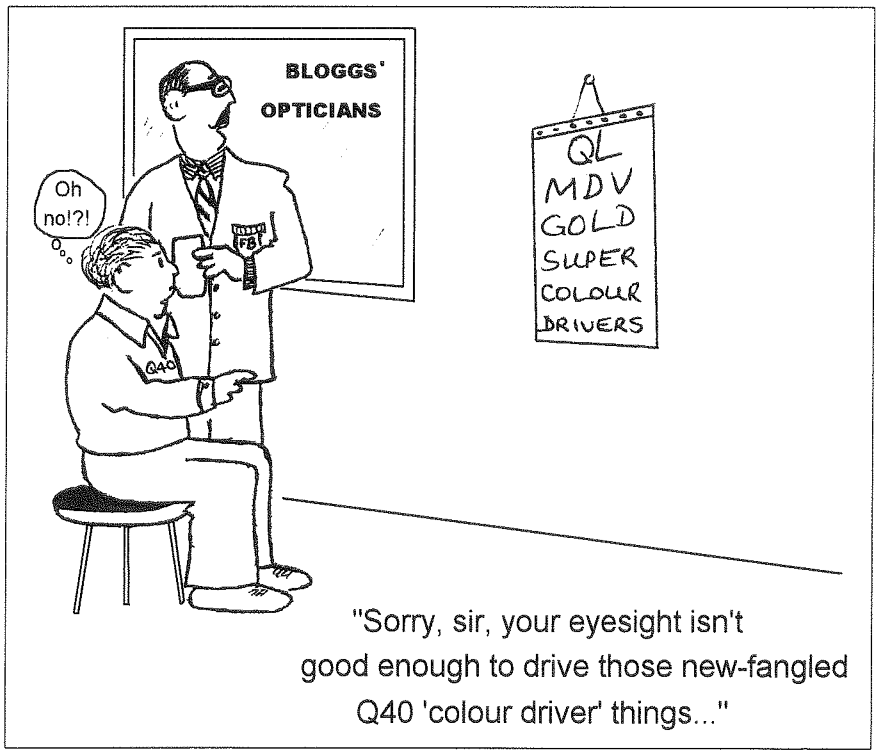
"Colour drivers", Email and internet access are probably the biggest and most important QL developments at the moment, but there are other significant activities going on as well.

Q-Celt Computing are hoping to bring out CD-ROMs of software for the QL scene (admittedly only for systems which can access CD-ROMs - mostly emulators). One QL user has a set of C source files for CD Filing Systems which may give hope of a QL implementation if help can be found from a suitably knowledgeable C programmer (contact me if you can help, I'll put you in touch).

Dave Westbury has been working on implementing JPEG graphics compression and Thierry Godefroy has ported a P.O.V. ray-tracing package to the Q40 initially - hopefully projects like this are a sign that graphical software development will really begin to happen once colour drivers are available. All this points to a bright new dawn for QL-compatible systems. We are seeing good news after good news at the moment and I hope it carries on.

The bad news is that we had promised a cover disk with this issue. I'm afraid we had to postpone it after discovering major problems at the last minute, and are working with the author concerned to try to fix the problems, so the cover disk and probably another little gift will hopefully arrive in time for Christmas with the next issue. We wish to apologise to our readers for having to delay the cover disk.

Cartoon



"Sorry, sir, your eyesight isn't good enough to drive those new-fangled Q40 'colour driver' things..."

News

Jochen Merz Software

Several customers requested it: a Secure Transaction Web page. Now it exists. Just follow the link under the "Comments and Orders" field and you will get to a Secure Transactions page (provided, your web browser supports it). Here you can enter credit card details without the need to worry, everything will be encrypted before it is sent over the internet. Not that I ever had a case of credit card fraud, but some people seem to dislike to send card details unprotected over the net.

QPC2 is being shipped, and it works very well. See the JMS advert in this issue for more details. Some nice facts to know: QPC2 also works on PowerMacs with PC emulators running Windows95 - it was tested on both SoftPC and RealPC - which means SMSQ/E is available for more or less every system now.

Wolfgang Lenerz (author of FiFi, WinEd etc.) will soon have a version of his new Agenda program - which will be distributed by Jochen Merz Software. The Agenda will run under the Pointer Environment (with the 'standard WMAN look') as well as under ProWesS - so you'll have the choice! Details in the next QL Today.

RWAP Software

As from September, the new address for Rich Mellor will be:
4 Anvil Crescent,
Coseley,
Bilston,
West Midlands
WV14 8GA
and the new telephone number will be **(01902) 83688**

Some of RWAP's software prices have been reduced and there are some updates to the various Q-Route maps.

Release 2 of the SBASIC/SuperBASIC Reference Manual is now available - updates cost £6 each or order 2 for £10. There is only one copy of the reference manual itself now left in stock - after this has gone, we shall only order more re-prints once several orders have been received.

Nasta'S Web Site

QL peripheral designer Zeljko Nastasic (better known to most of us as Nasta) has his own Web site, which was not listed in the Web sites article in the last issue. It includes information about and pictures of his various QL peripherals such as Qubide and Aurora, and some information about as yet unreleased boards such as an Ethernet card, CD-quality sound card and the Goldfire of course. The site also includes some information about himself, a photo and of course just how to pronounce his name! There's also links to information about his native Croatia. The address of the site is:
http://members.aol.com/nasta000/QL/ql.htm

POV Ray Tracer

POV (Persistence Of Vision ray-tracer) v3.1g is now available for QDOS /SMS systems from Thierry Godefroy's Web site:
http://www.imaginet.fr/~godefroy/english/download.html
This port supports Aurora 256 colours and Q40 65536 colour modes.

QBranch

1. We expect to release Mark Knight's Fractal Collection at the Paris show. This is a program to create and display various fractals many of which can be animated. These have been demonstrated at various QL shows this year. George Gwilt has provided many of the routines to run these programs.
2. We have a few new ED disks in stock. Price £15 for 10 boxed.
3. We also have a couple of complete tower cased Aurora system for sale. They consist of Aurora / Qubide / Super Gold Card / superHermes/ hard disk / floppy drives complete and working. Call for details.

EBOARD

Roy Wood has created an eboard for QL users. The idea is that it is an electronic notice board that any of you can post notes or announcements on. It may be good for people who don't want the major discourse that happens in email mailing lists or newsgroups, but want a quick checkpoint to see if there is anything new available (PD or commercial) or for private sales. This is an open board. To access it, go to
http://www.eboard.com
and enter SinclairQL in the box.
Entry password is: **qbranch**
Password for depositing messages is: **microdrive**
The board has only 1MB of space so keep messages short - don't include binaries or pictures. It would be especially useful for people to sell unwanted pieces of QL stuff or post a wanted advert.

Peter Jäger Site

A site which was not listed in last issue's guide to QL internet sites was Peter Jager's QL support site. Peter is the sysop of the Quasar BBS, and his site includes QL hardware information, software pages and so on. This is a well established site that is still being updated. Give it a visit at **http://ourworld.compuserve.com/homepages/peta**

Jonathan Hudson Site

Jonathan Hudson's Web site has now moved to **http://home.freeuk.net/diagon.alley/index.html**
Visitors to his old site will get automatically redirected to the new site after about 15 seconds. Jonathan can now be emailed on **jrhudson@bigfoot.com**

Q-Celt

Q-Celt Computing will be at the Byfleet Quanta workshop on October 3rd, and we are also hoping to organise a QL show in Northern Ireland, possibly in early December.
By the Byfleet workshop, we are hoping to release some CD-ROMs for QL emulator users. The CD-ROMs will contain the applications and data in a QXL.WIN file, so that QXL, QPC and QemuLator users for example can read them directly. Where possible, the CD-ROMs will also include copies accessible from DOS/Windows so that users without QXL.WIN access (e.g. Qubide users), but with access to a PC to transfer files, can make use of the CD-ROMs. It is not yet known which title will appear first, but it is hoped to release collections of Line Design clipart and fonts, QL screens clipart collection, a Literature collection and a QL PD/Shareware collection. We may also release our set of commercial QL software on CD-ROM if there is demand.
We are currently looking at producing a version of our database program, EasyBase which includes a simple password protection system for its own database files. You would be able to specify a 4 digit password for files you wish to protect from prying eyes (might be useful in an office environment in these days of sensitivity to security of information). Files which contained a password you specified would not load unless the correct password is entered (we would be able to recover your data for you if you happened to forget the password!). If you don't specify a password, files load normally, and files created with older versions of EasyBase remain compatible with the new version. What do you, the users, think - would this be a potentially useful new feature?
We are also working on a simple to use pointer driven diary program for the QL, based on a dated list of events and associated notes. Some work is still needed on this and we would welcome suggestions from users to incorporate into the program.

JPEG Decoder for QL Compatibles?

Acting Quanta editor Dave Westbury has revealed he is working on a machine code JPEG decoder to work on the Aurora in 256 colour mode, and the 16 bit 65,536 colour mode on the Q40. In mid September he had managed to get a trial version of JPG viewer software working.
JPG is a variable compression file format used for storing digital photographs on other computers such as PCs, and JPG files are common on the World Wide Web, and also used with some digital cameras, for example. The fact that compression ratio and picture quality can be traded against each other to some degree means that significant reductions in the file size of large graphics files are possible (compared to, say, GIF files) if you don't mind some loss of resolution or quality, although it is not yet known what features Dave Westbury will build into his JPG viewer.
He comments that the pictures viewed on the Aurora has "nice pastel shades", because of the lower number of colours, but the Q40 is better, despite the 2:1 aspect ratio.
At the time of writing this in mid September he told me "it's early days yet, I hope to have some sort of JPG carousel viewer in time for the Byfleet Quanta workshop."
dave.westbury@btinternet.com

W.N.Richardson & Co.

Issue 4 of Z88 User magazine is now available from Bill Richardson. Described as "A special internet and connections issue" it includes articles about modems, serial communications, connecting two Z88s, seeking out fax modems for the Z88, using the EMACS editor with a Z88 terminal, a News Roundup from Alchemist Research, a rundown of Z88 emulators on other computers and a list of additions and corrections for the Z88 User Manual 3rd edition. There is also a large Z88 circuit diagram at the back of the magazine.
Z88 User magazine is also looking for a new editor. Contact Bill Richardson if you are a Z88 user and fancy helping to produce the magazine.

OOPS...

Thierry Godefroy writes:
In the last QL Today issue, QLCF BBS phone number was wrong: it is +33 (0) 298 430 320 (and not 498 ...). Also, I was very surprised by the wrong statement saying than Phil Borman's BBS (Nene Valley) was closed: it is still up and running (this his also my hub and I poll him everyday).

QUANTA

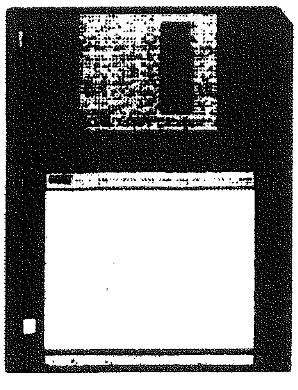
Independent QL Users Group

Worldwide Membership is by subscription
only, and offers the following benefits:
Monthly Newsletter - up to 40 pages
Massive Software Library - All Free !
Free Helpline and Workshops
Regional Sub-Groups. One near you?
Advice on Software and Hardware problems
Subscription just £14 for UK members
Overseas subscription £17

Barclaycard: Visa: Access: Mastercard

*** Now in our SIXTEENTH successful year ***

Further details from the Membership Secretary



Bill Newell
213 Manor Road
Benfleet
Essex
SS7 4JD
Tel. (01268) 754407

Stella

Arnould Nazarian

Here is a text that I have written in July 1999 to introduce the description of Stella as found on the Internet for over one year at http://wwwuser.imaginet.fr/~godefroy/stella/

Stella is the codename of an operating system that Tony Tebby proposes to develop if there were enough funds.

For those who do not read Quanta, this text was sent to Microsoft because I think that they have some problems with their new Windows for embedded products, named CE. Indeed nowadays they seem to buy market shares rather than sell the product, which means that they give away (invest) more money than they can invoice (my personal point of view, I do not have any information about the financial accounting of Microsoft!). I am persuaded that the Stella technology, even hidden behind the Windows API could help them a lot to improve their product. And it may help the computer science and technology to evolve a bit.

The text is reproduced as it was sent, with its errors and my French accent. But I have added references to notes at the end of the text to explain some things and to add ideas which came after the text was sent.

Stella: a user's point of view

Arnould Nazarian
64, boulevard de Pesaro
F-92000 Nanterre
arnazarian@magic.fr

0. Summary
1. Multitasking operating systems: a historic paradox
1.1. Microsoft Xenix and the Apple Lisa vs. MS-DOS and the Macintosh
1.2. The real time operating systems
1.3. Palm OS 1.4. EPOC

2. Multitasking operating systems kernels: 2 technical approaches
2.1. An important function of operating system kernels
2.2. The classic approach
2.3. A new approach

3. A comparison of the different technical approaches in commercial systems
3.1. Monotasking operating systems
3.2. Classic multitasking systems (and Linux)
3.3. Palm OS
3.3. EPOC
3.4. The Stella proposal

4. Conclusion

0. Summary

In the first part of this document there is a discussion about a historic paradox concerning the use and the (lack of) acceptance by the consumer market of true multitasking operating systems for personal computers, small personal digital assistants and the like. Indeed during the past 20 years the market did obviously choose to use strict or quasi monotasking environments, when the theory of multitasking operating systems had been quite stable for longer than 20 years and the advantages of using true preemptive multitasking time sharing environment seem so straightforward.

The second part of this document will first remind one of the fundamental service of an operating system kernel in a multi-tasking/-threaded environment. It will then try to show the behaviour of classic multitasking systems versus Palm OS and EPOC, two systems which seem to get acceptance by the mass market today, in relation to this fundamental service expected from an operating system. In order to be more comprehensible, this comparison will be made with the help of an analogy taken from the real world (car traffic). Thanks to this analogy it will be shown that only a very small number of operating systems having in common a way to achieve this service were successful in the mass market to date.

The third section will then do a second comparison between the operating systems available, but from a technical point of view based on the notion introduced in the second section.

In conclusion it will be reminded that EPOC and the proposed Stella operating system both result from the ideas of the same system programmer who started to study these subjects about 25 years ago.

1. Multitasking operating systems: a historic paradox

1.1. Microsoft Xenix and the Apple Lisa vs. MS-DOS and the Macintosh

The theory of the internal operations of multitasking (or multithreaded) operating systems is well known for more than 20 years. It first lead to the UNIX operating system as well as to a whole

range of real time operating systems less known by the public and mainly developped between 1975 and 1985.
Despite the existence of the theory of multitasking OSes and the existence of products based on these theories, the mass market did choose to reject the early attempts made by operating system vendors to sell multitasking environments. There are two very famous examples of this.
At the beginning of the 1980's Apple launched a true multitasking machine based on (at the time) advanced hardware and the windows/icons/mouse user interface which would have such a tremendous success a little later. A lot of money was thrown into the development of the Lisa. From its specifications it was a very attractive design. However it never became a success, to the extent that this project nearly killed Apple.
Fortunately, a few months later Apple launched what was to become an extremely successful personal computer, the Macintosh. Oddly enough the Mac was based on a strict monotasking operating system.
Moreover, a few years later, the cofounder of Apple launched the NEXT workstation, a kind of new Lisa machine with even more powerful hardware and again based on a true multitasking operating system derived from the classic theories about microkernels. It did not have much more success than the Lisa and it is nearly forgotten by now.
During the same period of time Microsoft did apparently not believe in the long term success of the MS-DOS system. Thus Microsoft tried to prepare the future with the launch of Xenix, its own version of UNIX, as well as the attempt to make a multitasking system out of MS-DOS. Neither projects were successes: Xenix disappeared relatively soon and the efforts to make a multitasking MS-DOS never materialised due to unsolved technical problems (?)
It is also true that Microsoft had to support the unexpected phenomenal success of the monotasking MS-DOS system in the mass market.
It can be argued that in those years hardware was not fast enough and too expensive to support true preemptive multitasking systems. While this is certainly true, the EPOC operating system, running today in comparable hardware and being very successful in a given segment of the market tends to demonstrate that there are other reasons as well.

1.2. The real time operating systems

While the mass market did choose without possible doubts 2 monotasking operating systems, numerous companies had developped dozens of real time operating systems for specialised industrial applications. The most well known are:
- OS9 - VRTX - PSOS - VxWorks - AMX - Lynx OS - Nucleus - OSE - ThreadX - QNX - Windows CE - ...
These operating systems are claimed to need only small amounts of memory, and to react within a definable time frame, possibly quickly, to internal or external events.
While a few years ago some of the mass market users (a) could have believed that the MacOS/MS-DOS duopoly would be broken by some of these real time OS vendors, they never achieved to spread out of their niche. And even within this niche they have a very tiny market share, as the vast majority of today's existing embedded applications were developped with no operating system at all.
A careful study of the different descriptions available about these real time systems show that they are all based on very similar algorithms, also similar to those used in eg. UNIX, Xenix, Lisa and Next OSes. Thus, apparently, a similar design of the real time OSes lead to similar very small acceptance by the market.

1.3. Palm OS

Palm OS is the operating system developped by Palm Computing at the beginning of the 1990's. It has a big success in the consumer mass market: the Palm digital organisers sell best and they have the reputation to be very reliable.
Without doubt the hardware of the Palm PDA's was developped with a lot of care of what users really want. Also the public release of free informations for applications developpers has made a lot to help increase the success of these machines.
However the design of the operating system is quite innovative: it is a multitasking non time sharing operating system (see details in next sections). While this can seem primitive and not very future proof, it is the clear recent choice of the consumers, and Palm OS is here to stay for many years.

›From a user's point of view, Palm OS is a very small step towards the direction of the proposed Stella operating system.

(a) In fact only I had believed (and maybe hoped) this...

1.4. EPOC

EPOC is the strict result of an English school of system programming. The Psion PDA's lauched as early as 1986 are based on EPOC (b) (which had different names in the meantime). More importantly EPOC was recently chosen by the main manufacturers of hand held telephones for their future new generation of wireless communication devices.

While there are certainly marketing reasons for the success of EPOC in the Symbian consortium, it is difficult to believe that such a success is not also based on technical reasons. Obviously Nokia, Ericsson, Motorola and Matsushita have enough software specialists to test both EPOC, Windows CE and even the other operating systems for embedded systems.

Indeed, EPOC is a time sharing multitasking operating system which is not based on the classic approach.

While there are strong assumptions that EPOC is a derivative of the system programmer's ideas who now proposes the Stella operating system, it shows its 15 years old design. EPOC is still better than the others who are based on 30 years old theories, but Stella can in turn be considered as a modern successor to EPOC.

2. Multitasking operating systems kernels: 2 technical approaches

2.1. An important function of operating system kernels

In a multi-tasking/-threaded operating system, one of the main problem is to manage accesses to communication data structures shared between all the threads inside a process, or between a number of processes.

In a preemptive multitasking system, if many threads need to access those data structures, there is the need for protection mechanisms in order to let the tasks which are in critical sections (ie. executing the code that accesses the shared data structure) safely finish their access. Otherwise tasks which are in critical sections may be preempted and leave the data structures in non defined states. This would clearly lead to application and even system crashes.

(b) Not so sure anymore, because I have found a web site dedicated to the first Psion Organisers in the meantime, and I had a quick look at their internal workings... But it helps to suppose that it happened that way.

The number of such accesses is generally huge, and for a given time frame it grows in relation with the increase of speed of modern processors.

2.2. The classic approach

The classic approach is to protect those accesses with semaphore/mutexes. With this scheme a thread which needs to access a shared data structure must test a variable named a mutex (or a semaphore).

If the mutex is free:
- the thread gets the mutex,
- it starts to execute the critical code,
- at the end of the access it must free the mutex.

If another thread with a higher priority gets a chance to be scheduled, then the thread which accesse the shared data structure gets preempted, and the data structure is no more available for all the other tasks which may need it. Some time later, not well defined, the execution of the thread that was suspended is resumed and it can continue the execution of the critical code.

If the semaphore is not free:
- the thread is automatically blocked on the semaphore by the operating system,
- another thread is selected to run, hopefully the one holding the semaphore but not necessarily,
- the blocked thread gets a chance to access the shared data structure later, but it is difficult to foresee when as there can be many task switches inbetween.

Of course there can be many threads blocked in the waiting queue(s) of the mutex.

There is an excellent real world analogy to this behaviour. It is car traffic. Let us suppose that:
- threads are cars,
- shared data structures are crossings,
- semaphores/mutexes are traffic lights.

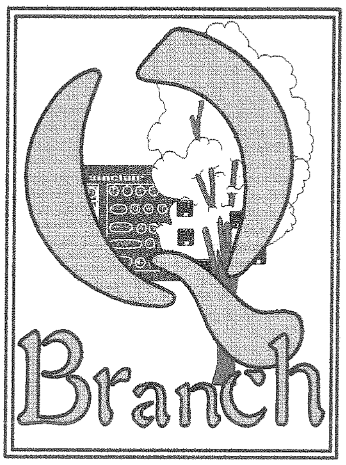
Then someone taking his car for a travel will encounter following situations at all crossings.

If the traffic light is green the car will have the permission to go on and pass through the crossing.

If a high priority car (police) must go from a given point to another one, it will do so somewhere else in the net of roads, without affecting our traveller, and if it happens to need the same crossing, then it may use it for a very short foreseeable frame of time.

If the traffic light is red:
- by law the car is automatically blocked on the traffic light, as well as possibly a queue of other cars,

Q Branch



Q Branch News

Shop hours :
Mon - Fri
10am - 6pm
Wed / Sat
10am - 2pm

This month sees the release of three new programs.
First off, from our own house, we have George Gwilt and Mark Knight s Fractal Collection. This is a stunning suite of graphics programs which demonstrate all of the power of the Mandlebrot and Julia sets and can be animated to produce eye-bending displays.
Geoff Wicks has also released QL2PC a program to convert text files from QL format into Rich Text Format and vice versa. this will allow you to transfer your files to and from a PC or other word processing package keeping most of the formatting intact. For a full list of features call us.
Finally we have the long awaited release of QPC 2. This is a completely new program which runs as a Windows 95/98 job and can be put to sleep on the taskbar and re-awakened just like any other windows program. Now you can have a QL anywhere you want.
At the Paris show we saw the first working colour drivers running on the Q 40. Still a bit of work to do but they look promising. Progress is also being made on the Internet software and when these two things are available Q Branch will have them for you !

Text 87
£ 79.00

Typset 94	£ 29.00
Fountext 94	£ 39.00
2488 drivers	£ 29.00
Epson ESC/P2 drivers	£ 26.00

Text 87 is the only QDOS / SMSQ wordprocessor capable of handliing the full screen on the Aurora / QXL / QPC systems. New drivers are currently being written.

UTILITIES

FiFi 2	£ 22.00
QSup	£ 32.00
QSpread v1.44	£ 66.00
Cueshell 2	£ 30.00
Qload / Qref	£ 15.00
Disk Mate 5	£ 16.50
QPAC 1	£ 20.00
QPAC 2	£ 40.00
QTYP 2	£ 30.00
QLQ	£ 32.00
LDUMP	£ 26.00
Q Count	£ 25.00

New IBM branded ED disks !
Supplied in a sealed plastic disk box - 10 disks with labels.
Buy now while stocks last !
only £ 15.00

'Just Words' by Geoff Wicks
THESAURUS, STYLE CHECK, SOLVITPLUS 3
£ 10.00 ea / ANY 2 PROGRAMS £ 18.00 / ALL 3 PROGRAMS £ 25.00
(Includes Pointer and non-pointer driven versions)
(P.E. versions need Hot_rext, WMAN and PTR_GEN or SMSQ/E to run)
Upgrades from previous versions £ 2.50 + S.A.E. New Manuals £ 1.50

QL2PC - New program !!
Convert text files from QL to PC formats
and much more !
Only £ 10.00
Spelling Crib : PD program £ 1.50 +SAE
or Free if you buy all three programs

PROGRAMMING

QD 98	£ 53.00
QD + QBasic	£ 69.00
QD + Qliberator + QBasic	£ 110.00
Qliberator	£ 50.00
Master Spy v 3.3	£ 30.00
QPTR	£ 37.00
Easyptr pt 1 & 2 (together)	£ 33.00
Easyptr pt 3 (C library)	£ 16.00
QMake	£ 18.00
QMon / JMon	£ 22.00
Basic Linker	£ 22.00
DISA 3	£ 37.00
QMenu	£ 16.00

Youv'e been asking for it and here it is !
The SBASIC / SuperBASIC Reference Manual
The complete definitive guide to BASIC programming in QDOS / SMSQ including three disks of PD toolkits, example procedures and an electronic index.
compiled by Rich Mellor, Franz Hermann and Peter Jaeger

Over 500 pages !
£ 40.00
+ postage

Q Branch

Tel +44 (0) 1903-200212 fax +44 (0) 1903-208070
email qbranch@qbranch.demon.co.uk web : http://www.qbranch.demon.co.uk

Q Branch

Feeling out on a limb ?
Reach out for Q Branch.
Suppliers of Quality QDOS/SMSQ products
Hardware and Software.

The Bank Volt, 6 Coronation Buildings
Ham Road, Worthing.
W. Sussex. BN11 2NN. UK.

ProWesS

ProWesS	£ 48.00
DATAdesign	£ 24.00
Fontutils	£ 30.00
File Search	£ 12.00
PFlist	£ 12.00
Fontpack	£ 60.00
LINEdesign v 2.16	£ 24.00
PWfile	£ 18.00

Paragraph

The ProWesS word processor

Demo version £ 1.50 + postage

Full Registered version £ 18.00

SMSQ/E

Gold Card / Atari / QXL Version

£ 76.00

Various Atari versions : call for details

The colour Drivers are coming ! I saw the prototypes at Paris and they will be available for the Q 40 soon.

QPC 2 is here !

£ 90.00 (£ 70.00 SMSQ/E Owners)

Upgrades from QPC 1 £ 30.00
(return master disk)

Special offer !
Get Cueshell for only £15.00 with any copy of SMSQ/E

Hardware

All new QXLs and Super gold Cards have now been sold. We do, however have some second user units available. We have two complete systems : Aurora / Super Gold Card / Qubide v 2.01 /superHermes / hard drive / twin HD drives / Tower Case.
We have other second user stock including a Super Gold Cards and a 4Mb QXL I.
Most of this stock turns over quite quickly.
Call us for details.

Recycled Gold Card	£ 60.00 *
Aurora	£ 90.00
Qubide	£ 55.00
Qplane	£ 25.00
Aurora cables	£ 3.00
Aurora rom adaptor	£ 3.00
'Arfa Braquet'	£ 8.00
'Son of Braquet'	£ 18.00
The 'Braquet'	£ 16.00
MC plate	£ 6.50

* when available.

14" and 15" monitors for the Aurora - Call.

Q Branch Programs

The Knight Safe 1 - standard version	£ 30.00
The Knight Safe 2 - with compression	£ 35.00
Q - Count Pointer driven home accounting	£ 25.00
Q - Route v1.07 Route finding programme	£ 25.00

The Fractal Collection !

Mandlebrot and Julia sets, Lorenz / Strange Attractors - discover the weird world of Fractals with this new program. Complete with many examples and routines to design your own screens.
Supplied on two disks - requires Gold Card and HD drives.
Hard drive preferable.

Only £ 35.00

Q 40

There are so many extras available for the Q 40 that it is hard to list them here. If you are interested in one of these boards than contact us and we will supply the details. the basic prices are shown here.

Q 40 board with 16Mb RAM & I/O card	£ 330.00
+ SMSQ/E	£ 30.00
Extra 16 Mb RAM	£ 30.00
Tower Case	£ 45.00

All prices do not include shipping.

VISA

We can accept payment by VISA, Mastercard and Switch. You can also pay by Eurocheques made out in Sterling or a Sterling cheque drawn on a UK Bank. Prices include Post and Packing in Europe.

MasterCard

- the other cars which have a green traffic light can use the crossing,
- the blocked cars know that they are blocked only for a given time frame. There can be many cars blocked in the waiting queue behind the traffic light.

Obviously there are many similarities between both situations. In both cases threads or cars know when they start, but it may be very difficult to foresee their time of arrival. Indeed mutexes normally free (resp. traffic lights normally green) may all be in use (resp. red) on a given day. And it is Murphy's law that says that the mutex will be in use (the traffic light red) when time is short...

However the mutex/semaphore system is worse than car traffic at least in one respect: it is usage that threads can be preempted in the middle of accesses to shared resources. Fortunately for car traffic, it is not considered normal that cars are stopped in the middle of crossings just because a high priority car needs some attention at the other end of the town. Think of the mess it would be in modern cities if a number of crossings were blocked because some drivers would have been interrupted by some more important task starting somewhere else! (c)

2.3. A new approach

The previous section, shows the major problem of the mutex/semaphore mechanism: it simulates an environment in which many threads/processes can run simultaneously in parallel with low interactions. In monoprocessor systems (eg nearly 100% of all installed systems) this model is very far from reality. Indeed threads never execute in parallel without interaction, as they execute strictly one at a time, and they strongly interact by having to share the processor's time.

(c) However in both cases, the same causes having the same effects, it is very difficult to foresee when the system will be overloaded. Once my wife asked me if I could explain why traffic jam occurs suddenly, only because the road net is crowded, but without any real reason like an accident or a crossing with a red light near the point we were. I only could answer that this is a very interesting problem for mathematicians. Now I guess that most of the multitasking computer systems work like car traffic, including traffic jam occuring because of mysterious reasons.
And believe it or no, it has taken me months, if not years, to find an easy to understand analogy between existing multitasking systems and the real world. In fact this analogy was just behind my windows (I mean my real windows!).

Let us now imagine a world with cars, crossings and red lights, but in this world, at any one time only one car may move for a few dozen meters under the supervision of a god, all the other cars having to wait for the attention of that god.

It seems obvious that in such a world, at the end of the allocated quantum of time to each car, the moving car would take a fraction of the time to park and free the road for the other cars to come. Moreover, if a car was in the process of passing a crossing just when its time quantum finishes, god would allocate a little bit more time for that car to finish its move and park. The result would be a net of roads and crossings always free for the unique moving car, no more need for traffic lights, and the cars coming next moving at full speed (there should be no pedestrians in that world!)

Why do most of the monoprocessor multitasking time sharing computer systems not work that way?

3. A comparison of the different technical approaches in commercial systems

3.1. Monotasking operating systems

This is the trivial case. As only one task will execute until the end of its execution, it can freely grab all the resources of the machine for itself without care.

There are two obvious advantages. The system's performance is improved as there is no operating system overhead while the single task is executing. And it is very easy to develop applications as the software developpers do not have to worry about protection of accesses to shared resources (there are no shared resources).

As was seen in the first part of this document, this approach was clearly chosen by the mass market at the beginning of the personal computer era. Above theory may not completely explain this very strange success of strict monotasking operating systems, but it could be a major reason for it.

3.2. Classic multitasking systems (and Linux)

Classic multitasking operating systems, those which are based on semaphore/mutexes for protection of shared resources are very difficult to use. Here a quote from a document published at
http://msdn.microsoft.com/library/techart/msdn_locktest.htm

« Multithreading is a little bit like Zen Buddhism: The more you learn about it, the more it escapes you. Although the Microsoft® Win32 multithreading application programming interface (API) is

fairly small (there are about a dozen functions altogether that deal with thread creation, manipulation, and synchronization), the hard part is to use it right. As I described in previous articles ("Synchronization on the Fly" and "Detecting Deadlocks in Multithreaded Win32 Applications"), the problems you can run into when not correctly using synchronization objects can be extremely hard to track down, and formal analysis methods cannot always be applied. »

Other eminent sources (Prof. Tanenbaum) state that it is as difficult to develop semaphore/mutex based multi-tasking/-threaded applications as it is to program in machine code.

And as was shown in paragraph 2, if there are many tasks, all of these tasks having to use many shared resources, then protecting the critical sections with mutexes but leaving those sections preemptible soon results in a mess.

This is maybe a major reason why these types of systems never got to get out of their niche of industrial applications: it may be because only engineers like this sort of difficulty (incidentally the same remark may be true concerning another well known and relatively successful operating system: Linux is a hacker's toy! It does not deserve more space here). (d)

3.3. Palm OS

From the programmer's manual it is very clear that:

- Palm OS is a multitasking operating system, in the sense that many tasks are always in the ready to run state,
- but Palm OS is not a time sharing multitasking OS, which means that when a task had started, it must continue until the end of its execution.

This is the reason why it is stressed on the fact that applications developped for Palm OS must stay short in order to not bother the user.

There is a scheme to break that otherwise quite rigid environment: any task may launch another task, which in turn may launch another task etc... It is the launching task's responsabitity (ie it's programmer's responsibility) to ensure that the resources needed by the chain of launched tasks are available when needed.

The net result is that Palm OS can not deal with long background operations (eg reorganisation of a database) even if the PalmPilot hardware is of the same family but more powerful than that of the first UNIX workstation at the beginning of the 1980's. Complex applications can be split into smaller parts though. Moreover Palm OS even does not allow simple immediate task switching. As long as the application's tasks are short enough, it gives only the illusion of task switching. It becomes obvious that Palm OS is a little more than a monotasking system, but it is very important to notice that from each running task's point of view, it behaves like a monotasking system.

Even if this environment can be considered as quite limiting it is the choice of the consumer mass market at the time of writing this presentation. Possibly the behaviour of Palm OS as explained in this document may help to understand this otherwise strange market choice.

Now comes the even more bizarre story behind Palm OS. Apparently, the original concept was to use a well known multitasking but classic kernel. As a matter of fact, Palm OS is a set of API's above the AMX real time kernel. What happened? Possibly the designers of the Palm Pilot did first choose to use AMX and its classic features:

- fixed priorities
- round robin scheduling at a given fixed priority
- shared resources protection with semaphores and mutexes

but then they might have decided to forget AMX in favor of their event driven Palm OS API because they thought that it would lead to a bug ridden and extremely difficult to program system, a system that the mass market may not accept?

The only remaining part of AMX in the Palm Pilot seems to be the TCP/IP communication stack which executes at a lower fixed priority than the user interface. Thus (inter)net communication seems to be suspended each time the user launches an application's task (this is at least the behaviour that can be gathered from reading the programmer's manual).

Otherwise the AMX kernel, still present under the Palm OS API, was carefully hidden from programmer's/user's access. Does this explain, at least in part, the success of the Palm Pilot range of personal digital assistants?

3.3. EPOC

EPOC has a very clear and different approach to time sharing preemptive multi-tasking/- threaded applications. It is best explained in the **EpocOverView.pdf** document which was available **http://www.symbian.com** in sections 3.2 to 3.6 and 4.3 and which was recently expanded into an .html document at

http://www.symbian.com/epoc/papers/active/active.html

(d) I am a little bit disappointed with Linux, but I may change my mind. Apparently I am not the only one. However I will not change my mind about Linux being transformed into Zen Bouddhism by mutexes/semaphores etc!

Semaphore mutexes are explicitely not supported because they are said to be difficult to use and dangerous.
Instead EPOC provides preemptive time sharing multitasking by the nearly forced use of so called RunL() functions which are not preemptible in order to access and use so called ActiveObjects. The atomicity of the RunL() functions is the method used to protect shared resources. And this scheme is so simple that the author of the document stressed on the fact that application programmers are most of the time not even aware that they enter a critical section accessing sensible shared resources.
As with PalmOS's applications, the author insists that RunL() functions must stay short. But as they are already at a level of granularity lower than that of applications in Palm OS, they have a very clear lead.
The net result is again that, apart from the overhead implied by RunL() function calls, applications most of the time run as if they were alone in the machine. In other words, from the point of view of applications executing under EPOC, the environment is mono-tasking like.
Coming back to the analogy with car traffic introduced in the previous section of this document, a tiny part of the time quantum allocated to each task/thread under EPOC is used to free "the roads" for use by the next car/task/thread
Could this be one of the explanations of the success of EPOC within the Symbian consortium? (e)

3.4. The Stella proposal

As can be seen in the Stella operating system overview written by Stella's author himself, the Stella proposal can be considered as a derivative of EPOC.
The main feature of the successful systems of the past and the present days is of course deeply embedded within Stella: a running task/thread, named "job" or "handler" also sees a mono-tasking like environment.
However the very big inovation of Stella are the INTSAFE intertask communication data structures. By design, INTSAFE data structures must not be protected. The most frequently used data structures are INTSAFE, an abbreviation of "INTrisically SAFE". (f)
Thus, while the main feature which made the success of monotasking operating systems, Palm OS and EPOC is still present in Stella, the need for special function calls and thus unproductive system overhead is mostly avoided. This is the key to the performance measured in prototype Stella systems. It will also be the key to its ease of programming as soon as it will have found a first niche market.

4. Conclusion

The author of Stella begun to work on these concepts many years ago. A smallish English company launched a product based on these concepts in the middle of the 1980's. This product can now be considered as a draft which has proven that the concepts work.
There are strong assumptions that Psion's EPOC is a derivative of these ideas, as it seems that the coworkers of this system's programmers were hired by Psion in 1986.
As was shown earlier in this document, this system programmer never gave up the work (while doing other things in parallel for a living). Today an emulator of the original system launched in the mid-1980's (but with many Stella ideas included) can be shown. Furthermore the most important routines of Stella exist and can be used for timing measurements.
Of course as the fundamental mechanisms in Stella are so clean, Stella also has other major advantages (eg. self cleaning configuration management, no possibility of priority inversions, memory manager immune from memory fragmentation...), that are developped by its author in another presentation document.
Lastly, one of the design aims of Stella had been from the beginning to foresee mechanisms to emulate the APIs of other operating systems. While the performance will not be as impressive as with a native Stella, existing systems may benefit a lot in terms of applications performances, ease of development and reusability of existing software components if their kernel was changed in favor of Stella and their API libraries adapted to Stella.
Stella is here to be choosen, adapted and used in the innovative digital products of tomorrow.
■

(e) As far as I understand, EPOC is neither so simple nor so clean. But it helps to describe it that way, as EPOC has taken 80 % of a market that Microsoft had hoped to grab and dominate like they did with the personal computer market.
(f) This is Tony Tebby's new innovation which he keeps for himself at the time being. In my opinion, the risk is that this is too innovative again to be accepted easily by the market. On the other hand it could be transparent to the application programmer, just like EPOC's active objects, and in that case it is a virtual winning technology.

SeekQL 2.09

Al Feng

SeekQL is a stand-alone, SuperBASIC database program which is intended to also complement the Archive-based DB-Easy (Wood and Wind Computing) front-end program. SeekQL is capable of writing and reading multiple files which have dedicated, user-defined and record-specific field labels; and like DBEasy, you can easily sWitch between files.

The main purpose of the SeekQL program is to create and access a simple names-and-addresses database using DB-Easy compatible "_exp" files (the truth is out there) -- you can either read existing DB-Easy "_exp" files using Seek-QL, or you can create DBEasy compatible "_exp" files.

In an effort to make the migration between the SeekQL and DBEasy database environments easier, the SeekQL program provides the user with a simplified, read-only representation of the DBEasy "single menu" record screen that I use.

The Program's History

Several years ago, I found myself relegated to using my unexpanded, backup QL. Despite the hardware limitations, the need for a database remained. Since Archive's programming language was too much for me to handle, I opted to write a simple, hierarchical, program ("Sbase") that wrote-and-read a simple "_txt" file. Quill was used for editing and printing address labels. Sbase's functionality was marginal; and I ended up relying more on Quill to both manipulate and read the file(s).

Although my hardware problems were resolved, the idea of enhancing the original Sbase program to also create "_DIF" (data interchange format?) files eventually led to the prototype for the "SeekQL" program. My brief sojourn in "Navajoland" interrupted thinking about and/or working on the program.

After the interlude, it occurred to me that the SeekQL program would have greater utility for me if it wrote/read DBEasy compatible "_exp" files.

Vestiges of the original program include the design of the "first" menu page and many of the PROCedure names. However, the "hotkey" concept employed by DBEasy is used in tandem with the "Function" keys ("F6" == "shift F1" & "F7" == "shift F2", etc).

Using the SeekQL Program

Before you can read any database, it must exist.

The directorY (press "F9" or "y") feature which will show you all of the files having an "_exp" suffix on the medium.

If you are NOT using an existing DBEasy compatible "_exp" file then you need to Create the file that you want to access. Of course, you will want to assign a meaningful filename (8 char. max) which suggests the nature of the data or use the default ("GADDRESS" == general address).

To sWitch the name, press the "F4" or "w" key and then INPUT the database's "new" name.

Next, you will want to establish the specific field names that will be dedicated to the "_exp" file by pressing either the "F7" (shift F2) or "s" key for the program's Screen edit function since this will make it easier to INPUT the data.

The field label file that is generated by the SeekQL program will have a "_lbl" suffix appended to the filename you have designated.

The "_lbl" file will be automatically loaded into the program when you Open an "_exp" file from the same storage device as specified for the "_exp" file. If the "_lbl" file is missing from the storage device, none will appear on the screen. Either copy the the "_lbl" file from a(ny) source, or "screen edit" a new set of labels.

SeekQL has no editing features beyond the initial creation of various files (nnnn_exp, nnnn_lbl & LineF_eed) which makes it particularly well suited for environments where the integrity of the database is a concern (e.g., a NETwork node).

The Print selection is set up for address labels, again "sharing" the format established within DBEasy; however, it is not set up to print the "country" field. The printer output can be readily extended to any "custom" output that you might be able to generate within DBEasy by modifying the "exp_to_txt" PROCedure (LINEs 2450 to 2840).

If you want to use the address label printing function as LLISTed, then the recommended labels for the first seven fields are:

```
Last Name
First Name
Address
more Address
City
State
Postal Code
```

As with DBEasy, these are only suggestions, and you can use whatever field labels that you choose.

Now, press either the "F2" or "c" key to Create the "_exp" file. Input data that corresponds to the labels you have defined.

As with DBEASY, the presumed "date" of the RECORD is the current date (presuming the QL's clock is correct). Whenever you have an active cursor,

you can INPUT whatever date-or-data you choose.
When you have finished putting in all the data for the final record in your database and the program presents you with a new RECORD screen, then INPUT "end" in the "date" field to close the file.
The "_exp" file you have Created can now be Opened for use directly by the SeekQL program or imported into DB Easy (or, Archive, alone).
Please note that if your ROM does not process WHEN ERRor properly, and the program halts (presuming you typed it in properly), then simply type "continue" to complete the Open process if it has balked. If the program still balks, then you will also need TK2_EXTensions (see below).
In addition to use of the arrow_keys for next/back/first/last, the user has the following options:

Find
More
Record
Print
sWitch
Exit

While SeekQL's functionality might seem limited, it includes the most used (by me, anyway) features of a database; and, the lack of extended features should make the program less intimidating and more accessible for the novice and yet provide a useful complement for the experienced (DBEasy, in particular) database user.
The Find and More search is case sensitive and defaults to lower case to provide a general search. Where appropriate, you can indicate an upper case search for proper nouns. Thus, using lower cases and the word "and" will provide find all instances of 'and', 'land', and other words containing the search string. On the other hand, using the upper case and the word "and" will not find 'land' but will find words such as 'Andrew', 'Andover', etc.
If you know the specific Record number, you can request it specifically.
As within DBEasy, you can sWitch between your "_exp" databases from the single record screen or from the main menu screen.
SeekQL operates in 80 column, monitor mode.

About the Program LLISTing

The program is rather long, but every attempt has been made to purge the really redundant statements.
Some statement clusters which might have been consolidated into a single PROCedure have not been to make "reading" the LLISTing a little easier.
SeekQL was designed to run on any QL -- from an unexpanded QL having only microdrives to a QXL (SeekQL's "search" and "display performance will be better with computers having faster processing speed). Well, that was the plan ...
The following LLISTing works well with either MINERVA or SMSQ. No TK2_EXTensions are necessary.
For reasons that are not clear to me, yet, several adjustments to the program have been necessary to make it work with the standard JSU ROM -- some of the "fixes" were minor, and some were major.
The minor changes included amending the width of BLOCKs to a maximum of only 511 (sedit, redit, Cmd_Line). The major changes relate to the on-screen display of the files, are still quirky (for example, 'commas' are not read to the screen) and are too lengthy to relate herein. Most serious in the "JSU" version is the fact that at the current time "find" and, "more" do not function.
If you have a JSU (or, JS) ROM, or equivalent, and feel adventurous, then you can attempt to amend the current LLISTing.
If you are using a 128K QL (i.e., with a MINERVA ROM), then you will have to change LINE 2590 to read:

`2590 DIM Z$(100,512)` or,
`2590 DIM Z$(72,640)`

The first number (100) indicates the number of RECORDS the SeekQL program will read, and the second number (512) indicates the maximum amount of data that will be read from each RECORD by the program.
You also need to change LINE 2610 to:

`2610 FOR c=0 TO 102` or,
`2610 FOR c=0 to 74`

These are just suggestions. If you have extensive data in each RECORD then you will probably want to maintain the second number at '756' and reduce the first number to '64' and the upper value in LINE 2610 to '66'. If you have extensive data in each RECORDS and you have more than 64 RECORDS then you are a candidate for memory expansion. For the time being, take advantage of the program's multi-file feature and split your data into more discrete files.
Of course, if your default storage device is not flp1_ then you also want to change LINE 210; for example, if you want to LOAD the program from mdv1_ then LINE 210 should read:

`210 Dvice$ = "mdv1_"`

Also, if you do not have "ram()_" devices, then references to "ram1_" should be changed to "Dvice$" for those PROCedures to function properly.
I have actually found that WHEN ERRor is impeded on the JSU and JS ROMS when

JOCHEN MERZ SOFTWARE

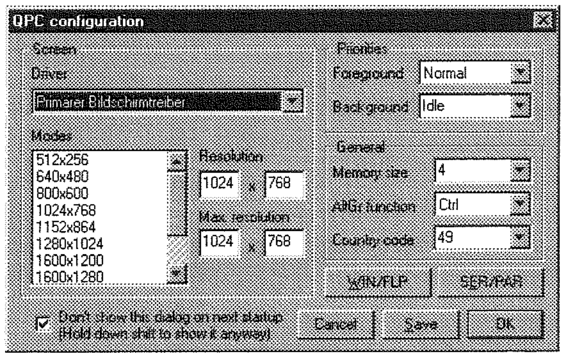
Im stillen Winkel 12 D-47169 Duisburg
Tel. 0203 502011 Fax 0203 502012
http://www.j-m-s.com/smsq/index.htm

QPC II exists!

We're not giving a full list of what QPC is: to make it short: it allows you to run the majority of QL Programs on PCs, and comes with the QDOS-compatible (but much faster and better) operating system SMSQ/E.

So what's QPC II then?

The more advanced version of QPC. Whereas QPC required your PC to run in DOS mode only while QPC was running, QPC II now allows you to multitask with Windows95, 98 and NT. You do not need to run your system down and up to start QPC II, just double-click it and it starts. Also, the configuration of QPC II is MUCH easier - it is done in a configuration window where all settings can be easily done. Installation is much easier too, all is done automatically with an installation program, or you just execute it directly from floppy disk or harddisk.
We have also reacted on the requests of many users: you can now have the QL windows scaled! For example, if you have a graphics card or a laptop with a resolution of 1024x768 pixels, then you can decide whether you want to use the full resolution (thus giving you 1024x768 pixels for the QL ... high resolution with small fonts) or if you want the 512x256 pixels to be scaled into 1024x768 (so that you have a large, nice original QL display). We discovered that using other devices is much easier too: if you have a laptop and a PCMCIA-modem (which acts as COM4, for example) then you can access it from QPC II as well!.


QPC configuration
Screen
Driver: Primarer Bildschirmtreiber
Modes: 512x256, 640x480, 800x600, 1024x768, 1152x864, 1280x1024, 1600x1200, 1600x1280
Resolution: 1024 x 768
Max. resolution: 1024 x 768
Priorities
Foreground: Normal
Background: Idle
General
Memory size: 4
AltGr function: Ctrl
Country code: 49
✓ Don't show this dialog on next startup (Hold down shift to show it anyway)

So, with QPC II you get
- ☞ the advantages of QPC
- ☞ plus much easier installation
- ☞ plus much easier configuration
- ☞ plus multitasking with Windows
- ☞ plus scaleable screen resolution

AT THE SAME PRICE AS BEFORE - only DM 249,-

You can upgrade to QPC II if you already own QPC by sending in your QPC master disk (don't worry, you get it back II with the latest version of QPC). The price is only DM 79,90.

To run QPC II, you need at least a 486 or Pentium, 16MB of RAM, VGA, Windows 95,98 or NT and DirectX.

TERMS OF PAYMENT

Postage and package **[Germany]** DM 8,99 (if total value of goods is up to DM 50,- then only DM 5,99). **[Europe]** DM 14,50 (if total value of goods is up to DM 50,– then only DM 9,50). **[Overseas]** between DM 14,50 (1 item) and DM 35,– (maximum). All prices incl. 15% V.A.T. (can be deducted for orders from non–EEC–countries). E&OE. Cheques in DM, £'s, Eurocheques and Credit Cards accepted.

TK2_EXTensions are invoked using a 128K QL. If your QL's ROM does not process WHEN ERRor properly then try changing LINE 2430 to:

```
2430 CONTINUE
```

As LISTed, the program uses my default for the labels that I use if-and-when the LineF_eed file is not found on the LOADing device. Once you ascertain the proper settings for the labels you use, you should input the appropriate data in LINE 2430.

SAVE the program on a fresh medium, RESET your computer, and see if the program LOADs. If the program still halts, then REMark LINEs 2420-2440 and 2630-2650 (i.e., the "WHEN ERRor" routines) and SAVE; but, do not omit these lines from the LISTing. Now, when the SeekQL program halts, type "continue" (and then, press the ENTER key) to proceed.

Of course, the "other" REMarked statements can be omitted. The REMarked DIM statement (LINE 80) is incomplete and has been included for reference purposes.

If you do not want to type in the program, the program is available from me for $10.00 ($10.00 is the requested SHAREWARE price -- in addition to the program, you will receive sample files, and other PLATYPUS Software programs). Don't forget to state the disk size you use, otherwise a 3.5" disk will be sent; microdrive users should also send two formatted micro-cartridges.

HAPPY TRAILS, AND COMPUTING, TO YOU ...

[The program listing will be on the next cover disk, remember that this is SHAREWARE!] ■

Gee Graphics! (on the QL?) - part 12

Herb Schaaf

Reflection and Shear in 2 Dimensions

Reflection is another way of transforming points; in GG#8 we looked at results (in 2D) of translation, scaling (zooming), and rotation transforms carried out in 3D. In GG#10 we used a stereo pair to help us see the effect of these 3 transformations in a "3D" illusion. Those three transforms did not change the shape of the graphic object.

Reflection changes the "handedness" of the object, and shear deforms the object.

Hopefully when you run the listing

`QL_SheaReflectLess3_bas`

you will be able to see the effects of 5 transformations, and next time we will add three more.

Reflection in a mirror changes the handedness of an object. If you can look directly at your left hand while looking at the reflection of your right hand in a mirror, you can turn them about and see that they appear as objects having the same shape and handedness. If you have 2 mirrors touching at right angles in a corner, looking at the two reflections will show you what you look like to others and so might be part of the gift that Robt. Burns would have the giftee gie us so that "we could see ourselves as others see us".

Reflection is often done with reference to the horizontal or vertical axis, but can also be with reference to any line. An odd number of reflections will change the "handedness" of the shape, an even number of reflections will not.

Reflecting a point simply puts it on the other side of a reference line and at the same distance from the line as before, such that the path that connects the original object point and the reflected image point is at right angles to the reference line. The reflection takes place along the shortest path from the object point to the line, crossing the line and then extending that same distance away on the other side of the line to the image point. The path is perpendicular to the line.

To define the object point we can use its x and y coordinates. To specify the reference line we can use the "general" form Ax + By + C = 0. If A = 0 then we have a horizontal line, if B = 0 we have a vertical line. If C = 0 the line goes through the origin 0, 0. If both A and B are zero then C = 0 no matter what and we don't have a line at all! Another form for the line is the "slope-intercept" form, y = mx + b, where m is the slope of the line, and b is the y value when x = 0 and the line crosses the y axis. Yet another form for the line is the "normal" form x * COS(alpha) + y * SIN(alpha) = p, where p is the distance from the origin to the line and alpha is the angle the line makes to the horizontal x-axis.

See the listing for the FuNction Point_to_line2D (x,y,A,B,C) that finds the shortest distance from a point to a line. The line is a theoretical line of infinite length, and not just a segment of line having two ends.

Shear is a transformation parallel to the reference line where we use the distance from the point to the reference line as a measure of how far the point is to be moved parallel to the reference line. A common example of shearing is the creation of "Italicized" font from a vertical font style.

The FuNction project_on_line (x,y,A,B,C) in the listing returns the 'image' values for x and y for both shear (x_s, y_s) as well as mirror (x_m, y_m) reflection.

The listing

QL_SheaReflectLess3_bas

takes the letters "Q L" from GG#1 and lets us to perform various 2D graphics transformations. To get the DATA for the Q_curls in the Q's curlicue I went back to

Show_ARCircle_bas

from GG#4. I was puzzled by the answers I got and found a problem in line 590 of that listing. The INT should have also been applied to the first term so as to have the same precision for both parts; line 590 should read:

"radius_angle=((INT(chord_angle)+90)MOD 360)+(chord_angle-INT(chord_angle))".

Another option seems to be to simplify line 590 to:

"radius_angle=chord_angle+90"

and forget all about the MOD and INT. It also seems that line 300 is not needed! What fun to go back in time and debug old programs; thank goodness for the REMarks!

As you can see by examining the listing, we first get the control points for the letters into arrays, then draw them. The letters are 90 units high and start out in the first quadrant. You are presented with a menu and choose by touching a letter key. Any numerical values must be entered, and the prompt should tell you what to do. For the shear and reflection operations you set the reference line to be horizontal, vertical, or sloping and then make adjustments, using the indicated keys. After transformation the new mapping is shown in red for comparison with the object that it was transformed from in black. We are now closer to having the "flying QL logo" that Bill Cable had asked for. I encourage you to put in other values for circ_parts (around line 170) just to see the effect. Small values make for clunky looking curves; large values take time and memory.

Another deforming transformation is selective scaling along a single axis to change the aspect ratio; next time we'll add that option and also illustrate projections and perspectives in 2D.

Since the listing for the next article with three more 'ects'is intended to be merged with the listing in this article the line numbers should be kept as they are.

```
100 REMark QL_SheaReflectLess3_bas for GG#12  H.L. Schaaf August 25, 1999
110 MODE 4 : WTV  : REMark 4 colors, window 448 wide, 202 high
120 vert_scale = 300 : graspix = 476/645
130 IF VER$=="JSU" THEN graspix = 344/549
140 horz_width = (vert_scale/(202-1))*(448-1)*graspix
150 SCALE vert_scale,-horz_width/2,-vert_scale/2 : PAPER 4 : INK 0 : CLS
160 large_number = 2^1000 : ref_line = 0 : circ_parts = 42
170 :
180 set_letter_arrays : REMark 3 arrays :  Q_oval, Q_curl, and L_legs
190 :
200 REPeat options
210  CLS : i_nk = 0 : INK i_nk : draw_letters : show_x_y_axes
220  get_options
230  IF (ref_line) : i_nk = 7 : show_reference_line
260  IF choice_made  THEN
270   i_nk = 2 : draw_letters : i_nk =  44 : show_x_y_axes
280  END IF
290  PRINT#0\\,,"touch [spacebar] for menu" : PAUSE
300  ref_line = 0
310 END REPeat options
320 :
330 DEFine PROCedure set_letter_arrays
340  DIM Q_oval(2,circ_parts,2) : REMark Q ovals
350  FOR i = 0 TO circ_parts
360 REMark 1 for outside of Q oval
370   Q_oval(1,i,1)= 40 + COS(PI*2*(i/circ_parts))*40*.8
380   Q_oval(1,i,2)= 50 + SIN(PI*2*(i/circ_parts))*40
390 REMark 2 for inside of Q oval
400   Q_oval(2,i,1)= 40 + COS(PI*2*(i/circ_parts))*30*.8
410   Q_oval(2,i,2)= 50 + SIN(PI*2*(i/circ_parts))*30
420  END FOR i
430 :
440 REMark 4 arcs (and 2 lines) for the Q curlicue
450 DIM curl_data(4,5) : REMark for the curlicue in Q
460 RESTORE 470
470  DATA 74.81073,  34.08263,  27.50658, 259.9275,  223.9275
480  DATA 29.61158, -10.77684,  36.18034,  45.43495,  81.43495
```

```
490  DATA 46.46447,   9.393398, 20.65857,  94.06505,  49.06505
500  DATA 82.07107,  44.14214,  29.21564, 220.9349,  265.9349
510 FOR i = 1 TO 4
520  FOR j = 1 TO 5 : READ curl_data(i,j)
530  FOR j = 4, 5 : curl_data(i,j) = RAD(curl_data(i,j))
540  curl_data(i,0) = curl_data(i,5) - curl_data(i,4)
550  curl_data(0,i) = curl_data(i,0) / (2*PI) * circ_parts
560  curl_data(0,0) = curl_data(0,0) + INT(ABS(curl_data(0,i))) + 1
570 END FOR i
580 curl_parts= INT(curl_data(0,0) / 4) + 1
590 curl_data(0,0) = (4 * (curl_parts) )+ 1
600 REMark with 4 sets of data, keep track with curl_count
610 curl_count = 0
620 :
630 DIM Q_curl(curl_data(0,0),2)
640 REMark get center x and y, radius, start_angle, end_angle
660  FOR curl_num= 1 TO 4
670   x_c = curl_data(curl_num,1) : y_c = curl_data(curl_num,2)
680   radius = curl_data(curl_num,3)
690   begin_at = curl_data(curl_num,4) : end_at = curl_data(curl_num,5)
700   FOR i = 0 +((curl_num==2) OR (curl_num== 4)) TO curl_parts
710    angle = begin_at + curl_data(curl_num,0) * i / curl_parts
720    Q_curl(curl_count, 1) = x_c + COS(angle) * radius
730    Q_curl(curl_count, 2) = y_c + SIN(angle) * radius
740    curl_count = curl_count+1
750   END FOR i
760  END FOR curl_num
770  curl_count = curl_count -1
780 :
790  DIM L_legs(7,2) : RESTORE 800 : REMark letter L
800  DATA  134,20 , 98,20 , 90,10 , 134,10
810  DATA   98,20 , 90,10 , 90,90 , 98,90
820  FOR i = 0 TO 7
830   FOR j = 1 TO 2
840    READ L_legs(i,j)
850   END FOR j
860  END FOR i
870 :
880 END DEFine set_letters_arrays
890 :
900 DEFine FuNction SGN(n)
910  RETurn (n>0)-(n<0)
920 END DEFine SGN
930 :
940 DEFine PROCedure draw_letters
950  INK i_nk : FILL 1
960  POINT Q_oval(1,circ_parts,1),Q_oval(1,circ_parts,2)
970  FOR i = 0 TO circ_parts : LINE TO Q_oval(1,i,1), Q_oval(1,i,2)
980  INK 4 : FILL 1 : POINT Q_oval(2,circ_parts,1), Q_oval(2,circ_parts,2)
990  FOR i = 0 TO circ_parts : LINE TO Q_oval(2,i,1), Q_oval(2,i,2)
1000  INK i_nk : FILL 1 : POINT Q_curl(curl_count,1),Q_curl(curl_count,2)
1010  FOR i = 0 TO curl_count : LINE TO Q_curl(i,1), Q_curl(i,2)
1020  FILL 1 : POINT L_legs(3,1), L_legs(3,2)
1030  FOR i = 0 TO 3 : LINE TO L_legs(i,1), L_legs(i,2)
1040  FILL 1 : POINT L_legs(7,1), L_legs(7,2)
1050  FOR i = 4 TO 7 : LINE TO L_legs(i,1), L_legs(i,2)
1060  FILL 0
1070 END DEFine draw_letters
1080 :
1090 DEFine PROCedure show_x_y_axes
1100  INK i_nk
1110  LINE -vert_scale,0 TO vert_scale,0
1120  LINE 0, -vert_scale TO 0, vert_scale
1130 END DEFine show_x_y_axes
1140 :
1150 DEFine FuNction Point_to_line2D (x,y, A, B, C)
1160  vert_line = 0 : horz_line = 0 : crosses_origin = 0 :is_line = 1
1170  IF NOT SGN(A) : horz_line = 1
1180  IF NOT SGN(B) : vert_line = 1
1190  IF NOT SGN(C) : crosses_origin = 1
1200  IF ((vert_line) AND (horz_line)) : is_line = 0 :PTL = 0
1210  IF (is_line) THEN
1220   IF (vert_line) THEN
1230    PTL = (-C/A) - x
1240   ELSE
1250    PTL = (A*x + B*y + C)/(SGN(B)*SQRT((A*A)+(B*B)))
```

TF Services

superHermes

A major hardware upgrade for the QL

All Hermes features (see below for list) PLUS full 19200 throughput on ser1/ser2 not affected by sound IBM AT keyboard interface (plus foreign drivers) // HIGH SPEED RS232 industry standard two-way serial port. 4800cps throughput (supergoldcard - qtpi - zmodem) at 57600bps // THREE low speed RS232 inputs (1200 to 30bps) - Driver for SERIAL MOUSE supplied. Other uses include RTTY/graphics tablet etc // THREE spare I/O lines (logic) with GND/+5V // Capslock/scrollock LED connector // Turbo/keylock connectors // 1.5k user data permanently storeable in EEPROM

All this on a professional board about twice the size of the 8049 co-processor it replaces

Cost (including manual/software)	**£90** (£92/£87/£90)
IBM AT UK layout Keyboard	**£22** (£24/£23/£27)
Serial mouse	**£11** (£13/£12/£14)
Capslock/scrollock LED	**£1** (£1.50/£1/£1.50)
Keyboard or mouse lead	**£3** (£3.50/£3/£3.50)
High speed serial (ser3) lead	**£4** (£4.50/£4/£4.50)

superHermes LITE

All Hermes features (see above) + an IBM AT keyboard interface only. Entry level superHermes.
Cost (incl keyboard lead)...**£53** (£55.50/£51/£53.50)

Minerva

MINERVA RTC (MKII) + battery for 256 bytes ram. CRASHPROOF clock & I^2C bus for interfacing. Can autoboot from battery backed ram. Quick start-up.

The ORIGINAL system operating system upgrade

OTHER FEATURES COMMON TO ALL VERSIONS

DEBUGGED operating system/ autoboot on reset of power failure/ Multiple Basic/ faster scheduler- graphics (within 10% of lightning) - string handling/ WHEN ERROR/ 2nd screen/ TRACE/ non-English keyboard drivers/ "warm" fast reset. V1.97 with split OUTPUT baud rates (+ Hermes) & built in Multibasic.

First upgrade free. Otherwise send £3 (+£5 for manual if requd). Send disk plus SAE or two IRCs

MKI...£40 (£41/£40/£43) **MKII...£65** (£66/£63/£67)

QL REPAIRS (UK only)

Fixed price for unmodified QLs, excl microdrives. QLs tested with Thorn-EMI rig and ROM software.

£27 incl 6 month guarantee

QL RomDisq

Up to 8 mbyte of flash memory for the QL

A small plug in circuit for the QL's ROM port (or Aurora) giving 2, 4 or 8 mbytes of permanent storage - it can be thought of as a portable hard disk on a card, and reads at some 2 mbytes per second. Think of it - you could fully boot an expanded QL, including all drivers/SMSQ etc off **RomDisq** at hard disk speed with only a memory expansion needed.

2 mbytes RomDisq	**£39** (£41£37/£40)
4mbytes RomDisq	**£65**(£66/£63/£67)
8 mbytes RomDisq	**£98** (£100/£95/£99)
Aurora adaptor	**£3** (£3.50/£3/£4)

MPLANE

A low profile powered backplane with ROM port

A three expansion backplane with ROM port included for RomDisq etc. Aurora can be fitted in notebook case and powered off single 5V rail - contact QBranch for details. Two boards (eg Aurora and Gold Card/Super Gold Card/Goldfire fixed to base. Suitable for Aurora (ROM accessible from outside) & QL motherboard in tower case. Specify ROM facing IN towards boards, or OUT towards back of case.

Cost.............................£34 (£36/£33/£35)

I2C INTERFACES

Connects to Minerva MKII and any Philips I^2C bus

Power Driver Interface **16 I/O lines with 12 of these used to control 8 current carrying outputs (source and sink capable)**
2 amp (for 8 relays, small motors)............ **£40** (£43/£38/£44)
4 amp total (for motors etc) .... **£45** (£48/£43/£50)
Relays (8 3a 12v 2-way mains relays (needs 2a power driver)...**£25** (£28/£23/£27)
Parallel Interface Gives 16 input/output lines. Can be used wherever logic signals are required..**£25** (£28/£23/£27)
Analogue Interface **Gives eight 8 bit analogue to digital inputs (ADC) and two 8 bit digital to analogue outputs (DAC). Used for temp measurements, sound sampling (to 5 KHz), x/y plotting**..... **£30** (£31.50/£29/£30)
Temp probe (-40°C to +125°C)........ **£10** (£10.50/£10/£11)
Connector for four temp probes....... **£10** (£10.50/£10/£11)
Data sheets..**£2** (£2.50/£2/£3)

QL SPARES

Keyboard membrane	**£24** (£25/£24/£27)
1377 PAL	**£3** (£3.50/£3/£4)
Circuit diagrams	**£3** (£3.50/£3/£4)
68008 cpu or 8049 IPC	**£8** (£8.50/£7.50/£9)
8301/8302 or JM ROM or serial lead	**£10** (£11.50/£10/£11)
Power supply (sea mail overseas)	**£12** (£17/£16/£21)

Prices include postage and packing (Airmail where applicable). Prices are: UK (EC/Europe outside EC/Rest of world). Payment by cheque drawn on bank with UK address, debit card/Mastercard/Access/Eurocard/postal order or CASH! (No Eurocheques). SAE or IRC for full list and details

24 Sep 99

VISA

29 Longfield Road, TRING, Herts, HP23 4DG
Tel: 01442-828254 **Fax/BBS: 01442-828255**
tony@firshman.demon.co.uk **http://www.firshman.demon.co.uk**

MasterCard

```
1260   END IF
1270  ELSE
1280   PRINT #0;"Null coefficients for A and B, can't be a line." :PAUSE
1290  END IF
1300  RETurn PTL
1310  RETurn vert_line
1320  RETurn horz_line
1330  RETurn crosses_origin
1340 END DEFine Point_to_line2D
1350 :
1360 DEFine PROCedure transform_array(array)
1370  FOR i = 0 TO DIMN(array)
1380   SELect ON choice_made
1390    = 1 : x = array(i,1)*cos_rot - array(i,2)*sin_rot
1400          y = array(i,1)*sin_rot + array(i,2)*cos_rot
1410          array(i,1) = x  : array(i,2) = y
1420    = 2 : array(i,axis_num) = array(i,axis_num) + translate_amt
1430    = 3 : FOR j = 1 TO 2
1440           array(i,j) = array(i,j) * zoom_factor
1450          END FOR j
1460    = 4 : x_cl = project_on_line(array(i,1),array(i,2),A,B,C)
1470          array(i,1) = x_s : array(i,2) = y_s
1480    = 5 : x_cl = project_on_line(array(i,1),array(i,2),A,B,C)
1490          array(i,1) = x_m  : array(i,2) = y_m
1880    = REMAINDER : PRINT #0;"What's next ? ":STOP
1890   END SELect
1900  END FOR i
1910 END DEFine transform_array
1920 :
1930 DEFine FuNction project_on_line (x,y,A,B,C)
1940  PTL = Point_to_line2D(x, y, A, B, C)
1950  IF (B) THEN
1960   IF (A) THEN
1970    m = -A/B : y_i = -C/B : yp_i = y - (-x/m) : slope_angle = ATAN(m)
1980    x_pt = (yp_i - y_i)/(m+(1/m)) : y_pt = (-1/m) * x_pt + yp_i
1990   ELSE
2000    m = 0 : slope_angle = 0 : y_i = -C : yp_i = -C
2010    x_pt = x : y_pt = -C
2020   END IF
2030  ELSE
2040   m = large_number  : slope_angle = PI/2 : y_i = y  : yp_i = y
2050   x_pt = -C : y_pt = y
2060  END IF
2070 REMark closest point on line is (x_pt, y_pt)
2080 REMark reflected mirror image point is (x_m, y_m)
2090  x_m = x_pt - (x - x_pt)          : y_m  = y_pt - (y - y_pt)
2100 REMark shear transformation maps to point (x_s, y_s)
2110  x_s = x + PTL * COS(slope_angle) * shear_amt
2120  y_s = y + PTL * SIN(slope_angle) * shear_amt
2130  RETurn x_pt  : RETurn y_pt
2140  RETurn x_m   : RETurn y_m
2150  RETurn x_s   : RETurn y_s
2160 END DEFine  : REMark FUNction project_on_line(x,y,A,B,C)
2170 :
2180 DEFine PROCedure show_reference_line
2190  PTL = Point_to_line2D(0,0,A,B,C)
2200  IF (vert_line) THEN
2210   x_ref1 = -C/A : y_ref1 = -vert_scale
2220   x_ref2 = -C/A : y_ref2 =  vert_scale
2230  END IF
2240  IF (horz_line) THEN
2250   x_ref1 = -vert_scale : y_ref1 = -C/B
2260   x_ref2 =  vert_scale : y_ref2 = -C/B
2270  END IF
2280  IF ((is_line) AND (NOT ((vert_line) OR (horz_line)))) THEN
2290   x_ref1 = -vert_scale : y_ref1 = (-C -(A * x_ref1))/B
2300   x_ref2 =  vert_scale : y_ref2 = (-C -(A * x_ref2))/B
2310   IF ABS(y_ref1)< vert_scale THEN
2320    y_ref1 = -vert_scale : x_ref1 = (-C -(B * y_ref1))/A
2330    y_ref2 =  vert_scale : x_ref2 = (-C -(B * y_ref2))/A
2340   END IF
2350  END IF
2360  INK i_nk :LINE x_ref1, y_ref1 TO x_ref2,y_ref2
2370 END DEFine show_reference_line
2380 :
2390 DEFine PROCedure get_options
```

```
2400  REPeat query
2410   CLS#0 : choice_made = 0
2420   PRINT#0,,,"  [R]otate",,"[T]ranslate",,"[Z]oom"
2430   PRINT#0,," [S]hear",,"[M]irror reflection"
2450   PRINT#0,,,"[O]riginal letters",,"[Q]uit"
2460   transform$ = INKEY$(-1)
2470   IF transform$ == 'o' : RUN
2480   IF transform$ == 'q' : CLS#0 : STOP
2490   IF transform$ == 'r' : choice_made = 1
2500   IF transform$ == 't' : choice_made = 2
2510   IF transform$ == 'z' : choice_made = 3
2520   IF transform$ == 's' : choice_made = 4
2530   IF transform$ == 'm' : choice_made = 5
2570   IF choice_made : EXIT query
2580  END REPeat query
2590  SELect ON choice_made
2600   = 1 : rotate_menu
2610   = 2 : translate_menu
2620   = 3 : zoom_menu
2630   = 4, 5 : Set_reference_line
2660   = REMAINDER : GO TO 2400
2670  END SELect
2680 END DEFine get_options
2690 :
2700 DEFine PROCedure transform
2710  transform_array Q_oval(1)   : transform_array Q_oval(2)
2720  transform_array Q_curl      : transform_array L_legs
2730 END DEFine transform
2740 :
2750 DEFine PROCedure rotate_menu
2760  CLS#0 : INPUT #0;"ENTER degrees of rotation  ",rot_angle
2770  cos_rot = COS(RAD(rot_angle)) : sin_rot = SIN(RAD(rot_angle))
2780  transform
2790 END DEFine rotate_menu
2800 :
2810 DEFine PROCedure translate_menu
2820  CLS#0 : PRINT #0;"Translate along [X] or [Y] axis ?"
2830  axis$ = INKEY$(-1): axis_num = (CODE(axis$)-87) MOD 32
2840  IF axis_num > 2 OR axis_num < 1 : GO TO 2820
2850  INPUT#0;"ENTER translation along "&axis$&" axis ?",translate_amt
2860  transform
2870 END DEFine translate_menu
2880 :
2890 DEFine PROCedure zoom_menu
2900  CLS#0 : PRINT#0;"a factor greater than 1 will enlarge the object"
2910  PRINT#0;"a factor less than 1 will reduce it "
2920  PRINT#0;"negative values are OK, but Zero collapses all to a point"
2930  INPUT #0;"ENTER zoom factor : ",zoom_factor
2940  transform
2950  CLS#0:PRINT#0\\,,"Zoom factor was ";zoom_factor
2960 END DEFine zoom_menu
2970 :
2980 DEFine PROCedure Set_reference_line
2990  shear_amt = 1   : ref_line = 1
3000  CLS#0 : PRINT #0;"Choose a Reference line"
3010  PRINT #0;"[V]ertical, [H]orizontal, or [S]loping ?"
3020  Line_type$ = INKEY$(-1)
3030  IF Line_type$ == 's' : set__slope
3040  IF Line_type$ == 'v' : A = 1 : B = 0 : set__vertical
3050  IF Line_type$ == 'h' : A = 0 : B = 1 : set__horizontal
3060  valid_key = (Line_type$=='s' OR Line_type$=='v' OR Line_type$=='h')
3070  IF NOT(valid_key) : GO TO 2990
3080  IF transform$ == 's' : INPUT#0;"ENTER amount of shear ",shear_amt
3120    transform
3140  ref_line = 1  : i_nk = 2 : show_reference_line
3150 END DEFine Set_reference_line
3160 :
3170 DEFine PROCedure set__slope
3180  CLS#0 : slope = PI/4 : A =1: B=1 : C=0
3190  xo = 0 : yo = 0 :INK 242 : OVER -1
3200  PRINT#0;"Use CONTROL key with arrow keys for slope"
3210  PRINT#0;"move line up, down, left, right with arrow keys "
3220  PRINT#0;"use with SHIFT for finer adjustments; ";
3230  PRINT#0;" when satisfactory, touch ENTER";
3240  REPeat setting_slope
3250   p1x = -COS(slope)*1000 + xo : p1y = -SIN(slope)*1000 + yo
```

```
3260   p2x =  COS(slope)*1000 + xo : p2y =  SIN(slope)*1000 + yo
3270   LINE p1x, p1y TO p2x, p2y
3280   AT#0,3,0 :PRINT#0;"X=";xo;"  Y=";yo;
3290   PRINT#0;"  Slope = ";TAN(slope);
3300   PRINT#0;"  Angle = ";DEG(slope);CHR$(186);"           ";
3310   change_slope = CODE(INKEY$(-1))
3320   LINE p1x, p1y TO p2x, p2y
3330   SELect ON change_slope
3340    = 202, 218 : slope = slope - .1
3350    = 206, 222 : slope = slope - 1E-2
3360    = 194, 210 : slope = slope + .1
3370    = 198, 214 : slope = slope + 1E-2
3380    = 208 : yo = yo + 10
3390    = 212 : yo = yo +  1
3400    = 216 : yo = yo - 10
3410    = 220 : yo = yo -  1
3420    = 200 : xo = xo + 10
3430    = 204 : xo = xo +  1
3440    = 192 : xo = xo - 10
3450    = 196 : xo = xo -  1
3460    =  10 : EXIT setting_slope
3470    = REMAINDER  : GO TO 3270
3480   END SELect
3490   IF (slope < -2*PI) : slope = slope + 2*PI
3500   IF (slope >  2*PI) : slope = slope - 2*PI
3510  END REPeat setting_slope
3520  OVER 0 : m = TAN(slope) : y_i = (p1y+p2y - m*(p1x+p2x))/2
3530  x_i = -y_i/m : A = 1 : B = -A/m : C = -x_i  : CLS#0
3540 END DEFine set__slope
3550 :
3560 DEFine PROCedure set__vertical
3570  A = 1 : B = 0 : C = 0 : xo = 0  : INK 242 : OVER -1 : CLS#0
3580  PRINT#0;"change position with left or right keys"
3590  PRINT#0;"use with shift for finer adjustments"
3600  PRINT#0;"when satisfactory, touch enter"
3610  REPeat adjust_x
3620   LINE xo,-vert_scale TO xo,vert_scale
3630   AT #0,3,0 : PRINT#0;"X-axis intercept = ";xo;"    "
3640   change_x = CODE(INKEY$(-1))
3650   LINE xo,-vert_scale TO xo,vert_scale
3660   SELect ON change_x
3670    = 192 : xo = xo - 10
3680    = 196 : xo = xo -  1
3690    = 200 : xo = xo + 10
3700    = 204 : xo = xo +  1
3710    =  10 : EXIT adjust_x
3720    = REMAINDER  : GO TO 3620
3730   END SELect
3740  END REPeat adjust_x
3750  C = -xo : OVER 0 : CLS#0
3760 END DEFine set__vertical
3770 :
3780 DEFine PROCedure set__horizontal
3790  A = 0 : B = 1 : C = 0 : yo = 0   : INK 242 : OVER -1  : CLS#0
3800  PRINT#0;"change position with up or down keys"
3810  PRINT#0;"use with SHIFT key for finer adjustments"
3820  PRINT#0;"when satisfactory, touch enter"
3830  REPeat adjust_y
3840   AT #0,3,0 : PRINT#0;"Y-axis intercept = ";yo;"    "
3850   LINE -vert_scale,yo TO vert_scale,yo
3860   change_y = CODE(INKEY$(-1))
3870   LINE -vert_scale,yo TO vert_scale,yo
3880   SELect ON change_y
3890    = 216 : yo = yo - 10
3900    = 220 : yo = yo -  1
3910    = 208 : yo = yo + 10
3920    = 212 : yo = yo +  1
3930    =  10 : EXIT adjust_y
3940    = REMAINDER : GO TO 3840
3950   END SELect
3960  END REPeat adjust_y
3970  C = -yo : OVER 0 : CLS#0
3980 END DEFine set__horizontal
3990 :
5110 REMark end of listing QL_SheaReflectLess3_bas
```

Adventures on the QL

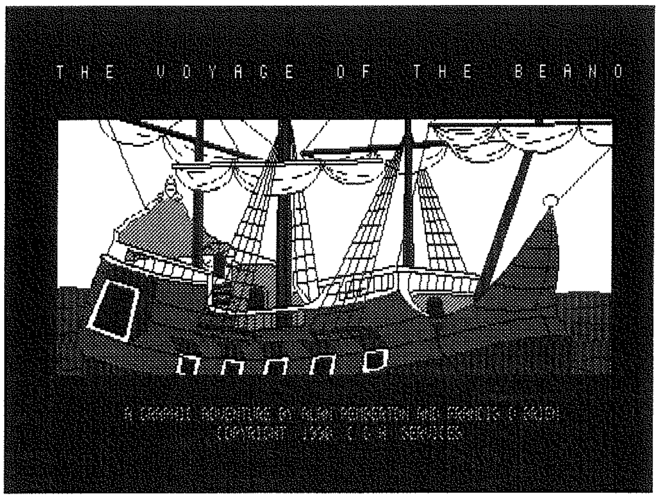
Part 4 - Voyage of the Beano

Darren D. Branagh

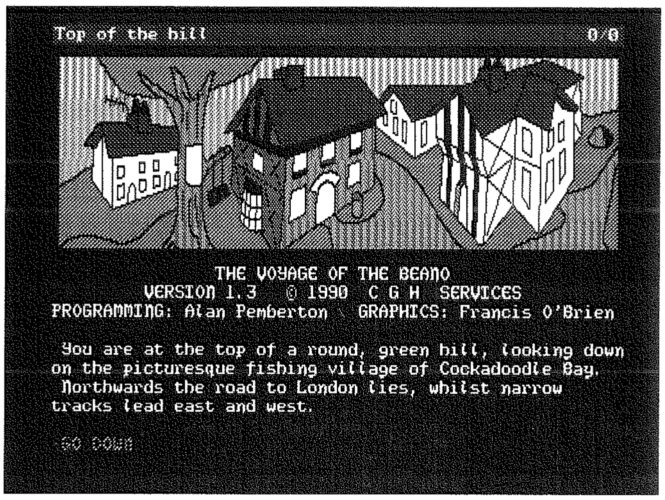
As promised in the last issue, this time we'll look at our first graphic adventure game. A graphic adventure is simply a text adventure with some graphics thrown in to liven up the gameplay, usually of the surroundings or any awful hazards or nasties you may be facing, they help to make the game more fun to watch and play, especially for kids.
I've chosen to look at a game called Voyage of the Beano, a long time favourite of mine and one of the games that really impressed me on the QL, given its sometimes limited graphic capabilities.
The game is available from the QUANTA library or via Ron Dunnett of Qubbesoft or Phil Jordan's soon to be available library. It is part of a collection of adventures written by Alan Pemberton, formerly of the Scottish QL User Group, and collectively named ADVENTURE '93, and if bought from Qubbesoft (as I did) is supplied on 2 DD disks. There are 5 separate adventures in the set, 4 smaller games of varying complexity on one disk (I'll be looking at one of these next time) and the 2nd disk contains solely The Voyage of the Beano, which shows what a large and visually great game it is.
The graphics (programmed by Francis O'Brien) are really good, and even Rich Mellor is mentioned in the credits (Hi Rich!) as a bug tester - a big game with a big programming panel too!

THE SCENARIO

The story of the game is this. You play the part of J.D. Hogwash, a former lowly deckhand in Her Majesty's service, who through a stroke of luck



(namely conning everyone into thinking he captured a Spanish galleon by himself) he finds himself the Captain of the H.M.S. Beano, with a mission to beat the Spaniards to the Gold. You are in total command of dear ol' J.D. and its your job to aid him in his task as he sets sail for Cockadoodle Bay.
The Voyage disk boots from its own BOOT program, and soon a large picture of an old sailing ship of long ago is displayed, along with a few bars of the Sailors Hornpipe, in glorious QL BEEP quality!!
Pressing any key starts the adventure, and off we go.



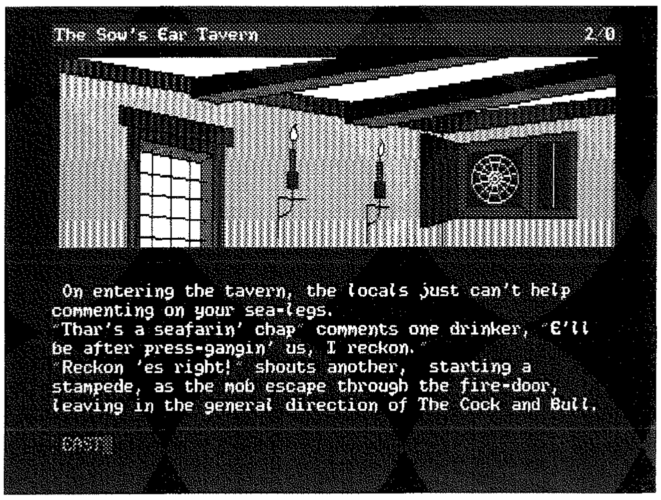
```
             THE VOYAGE OF THE BEANO
        VERSION 1.3   © 1990  C G H  SERVICES
PROGRAMMING: Alan Pemberton   GRAPHICS: Francis O'Brien

 You are at the top of a round, green hill, looking down
on the picturesque fishing village of Cockadoodle Bay.
 Northwards the road to London lies, whilst narrow
tracks lead east and west.

 GO DOWN
```

STARTING OFF

The screen is split into two areas of gameplay - The Graphics Window at the top of the screen and the Text Window at the bottom. The first graphic displayed shows Cockadoodle Bay in all its glory, and the first thing you realise is how good the graphics are - in QL terms, they are stunning, and it seems like more than the available MODE 4 colours are used.
The font used for the text is equally impressive, as its in an old celtic style font which suits the game somewhat. The directions and instuctions you enter are also displayed in the font. The descriptions are quite lengthy, leading to 'More...' being displayed on the screen, prompting you to press a key for the next screenful of text.

GAMEPLAY

Overall, gameplay is good, but as this is a graphic adventure, gameplay can be a little slow at times as the graphics have to be redrawn each time you revisit a location, which can take a few seconds on slower QL hardware, as the graphics are not loaded in full at boot, but are loaded from the disk as needed. This didn't



```
The Sow's Ear Tavern                                  2/0

 On entering the tavern, the locals just can't help
commenting on your sea-legs.
"Thar's a seafarin' chap" comments one drinker, "E'll
be after press-gangin' us, I reckon."
"Reckon 'es right!" shouts another,  starting a
stampede, as the mob escape through the fire-door,
leaving in the general direction of The Cock and Bull.

 EAST
```

affect me when playing 'Beano' on my Pentium III under QPC2, or on my QXL, as I was able to copy the game to my hard disk and run it from there which was much faster, but on a Trump Card with no hard drive which I later tested it on, it was a little annoying at times. It also means that the game disk must remain in the drive at all times when you are playing it, unless you copy the entire game everytime to a ramdisk and run it from there, which should work. However, on reading the accompanying manual (supplied on disk in a BEANO_DOC file, which runs to about 10 pages) I found this too has been overcome by the author.

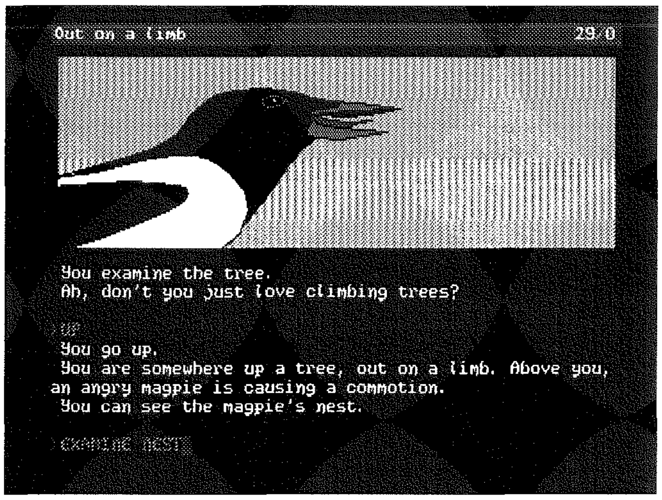
He has written an excellent couple of commands into the game, called GRON and GROFF. These are most useful if you have slower QL hardware, as they allow the Graphics to be switched on and off at will. Therefore, when you have visited a location once, and seen the graphics, you can just enter GROFF which switch the graphics display off. Should you visit a new location later, just GRON to see the pics!

Similar to a previous adventure we reviewed, PRON and PROFF do the same thing with a printer attached, should you wish to keep track of the layout of the game, and maybe draw up a map. Another nice feature here is a couple of extra commands too - VERBOSE and BRIEF. These are useful in case you feel the descriptions are too long and therefore taking up lots of paper when you print them, which can be costly. typing BRIEF sends a shortened description to the printer, basically just the nescessary facts. VERBOSE returns to the normal description again. Both again very useful.

SAVE and RESTORE both save and load games to disk under a filename of your choice, with the extension '_adv' so you can easily locate them, whereas STORE and RECALL do the same thing to memory in a ramdisk, which is quicker to do obviously, but you will need to SAVE these properly should you wish to switch off the QL, otherwise you'll loose your game.

Apart from the above, all the usual commands are accepted, such as QUIT and HELP, which are both self explanatory. Instructions can be simple commands such as GET SHOVEL or KICK DOOR, or you can string them together thanks to the parser used, such as GET KNIFE AND GO SOUTH THEN EAST. You can also interact with the people in the game and sometimes have conversations with them, a part I love. For example, ASK BUTCHER TO GIVE BONE TO ROVER or TELL ROVER TO SIT both work, or even ASK PIRATE ABOUT MAP, as does ASK MUGGER TO DROP KNIFE, though I doubt it would in real life!

I think this game is now PD, as it is offered by Qubbesoft and QUANTA, despite the _DOC file saying it is still copyright and sold by CGH Services. I think this is just a case of the instructions not being updated since the writing of the game, as it was formally commercial software but was declared PD when CGH ceased trading.



```
Out on a limb                                        29/0

 You examine the tree.
 Ah, don't you just love climbing trees?

 UP
 You go up.
 You are somewhere up a tree, out on a limb. Above you,
an angry magpie is causing a commotion.
 You can see the magpie's nest.

 EXAMINE NEST
```

I have been playing this game for years on my QL, and enjoy it a lot. It was one of the first PD disks I bought, and I still use it today. Despite being written in 1990, The Voyage of the Beano has stood the test of time well. It is very well written, by one of the best adventure programmers ever for the QL.

I am probably a little biased towards this game, as it is one of my personal favourites. However, this is perhaps its best endorsement - I own adventures for Ataris, Amigas, Spectrums, and even ZX81's, not to mention PC's, and this ranks along with the best offerings of those computers, many of which were more or less solely gaming machines, which the QL wasn't.

You'll enjoy Playing Voyage of the Beano - give it a try.

JOCHEN MERZ SOFTWARE

Im stillen Winkel 12 D-47169 Duisburg
Tel. 0203 502011 Fax 0203 502012
http://www.j-m-s.com/smsq/index.htm

Ink Attack

You can probably imagine how much ink was used up by printing several hundred colour stickers for the cover page of QL Today, so I had to inquire about getting ink at reasonable prices. Not only that the prices are pretty good, ink for new printer models is available too!

High-quality alternative ink cartridges for EPSON

Stylus Color 600, 800, 1520	Color	DM 21,90
Stylus Color 800, 1520	Black	DM 15,90
Stylus Color 500	Color	DM 19,90
Stylus Color 400, 500, 600, Photo	Black	DM 15,90
Stylus Color 300	4-Color	DM 22,90
Stylus Color Photo, 700, EX	5-Color	DM 27,90
Stylus Color 440, 640	Black	DM 17,90
Stylus Color 740	Black	DM 17,90
Stylus Color 440, 640, 740	Color	DM 23,90
Stylus 800 (old model)	Black	DM 12,90
Stylus Color II, IIs, 825, 1500	Color	DM 18,90
Stylus Color II, Iss, 825, 200	Black	DM 13,90

High-quality alternative ink cartridges for CANON

BJC-70, BJC-80	Black	DM 8,99
BJC-70, BJC-80	Color	DM 10,99
BJ 300, 330	Black	DM 11,99
BJC 600, 610, 620		
Black, Cyan, Magenta, Yellow	EACH	DM 7,99
BJC 800, 820, 880		
Black, Cyan, Magenta, Yellow	EACH	DM 11,99
BJC 4000, 4100, ...	Black	DM 8,99
BJC 4000, 4100, ...	Color	DM 12,99
BJC 7000 Series	Color	DM 32,99
BJC 7000 Series	Photo	DM 36,99
BJC 7000 Series	Black	DM 34,99

Other printing stuff

CD-Labels (10 pages with 3 labels) white	DM 6,99
Inkjet paper for DOUBLE-SIDED printing A4, 360 dpi, 80g, 200 sheets	DM 12,99
Inkjet paper for DOUBLE-SIDED printing A4, 720 dpi, 250g, 50 sheets	DM 22,99
Inkjet paper single-sided printing A4, 1440 dpi, 105g, 50 sheets	DM 17,99
Inkjet glossy paper A4, 1440 dpi, 10 sheets	DM 14,99
same as above, but 50 sheets	DM 49,99
Inkjet photo paper A4, 1440 dpi, 160g, 10 sheets	DM 18,99
Inkjet glossy paper for DOUBLE-SIDED printing A4, 360 dpi, 135g, 100 sheets	DM 18,99
same as above, but 100g, 100 sheets	DM 18,99

Minimum order

4 Items or DM 80,-

TERMS OF PAYMENT

Postage and package **[Germany]** DM 8,99 (if total value of goods is up to DM 50,- then only DM 5,99). **[Europe]** DM 14,50 (if total value of goods is up to DM 50,- then only DM 9,50). **[Overseas]** between DM 14,50 (1 item) and DM 35,- (maximum). All prices incl. 15% V.A.T. (can be deducted for orders from non-EEC-countries). E&OE. Cheques in DM, £'s, Eurocheques and Credit Cards accepted.

CAUTION: HOTKEYS - Don't Burn Your Fingers Part 1

David Denham

I should have known, give a magazine editor an inch and he'll take a mile.
After the MY BOOT article in the last issue and saying I wasn't that much of a hotkeys fan, Mr Jones lays down the gauntlet and before I know it I'm writing about hotkeys.

Actually, I'm quite glad he did, because it encouraged me to make the effort and teach myself the little bits I didn't know or had forgotten. So now I'm feeling more confident in their use. And ready to try to share my experiences with you. You have been warned - before you know it, I'll make a hotkeys addict out of you!

Let's start with what I would regard as a dictionary definition of hotkeys.
"A key definition which when pressed gives rise to an action which is independent of the task currently being done."
Oops, that's me in trouble straight away. Can someone transate it into words of one syllable for me please?
Before we go on, I must say that to use hotkeys, you need to have the pointer environment, specifically the PTR_GEN, WMAN and HOT_REXT files you get with QPAC2 and other pointerised programs. Toolkit 2 helps, but you can probably manage without it - most QL systems I have ever seen apart from an unexpanded QL seem to have it in one form or another.
The simplest program to install the minimum software needed to experiment with hotkeys is shown below:

```
100 TK2_EXT : REMark some systems need this
110 LRESPR FLP1_PTR_GEN
120 LRESPR FLP1_WMAN
130 LRESPR FLP1_HOT_REXT
140 HOT_GO
```

Those of you who have got this far, be patient, I'm just going to explain a little about pointer environment itself for those who have never used it.
The TK2_EXT command is used on some systems to activate Toolkit 2, so that we can use the LRESPR command. Otherwise you'd have to find the file lengths of the three files and have separate RESPR, LBYTES and CALL statements for each one, so the LRESPR command saves us a little bit of work by "automating" it to some extent.
PTR_GEN is the pointer interface; it gives you a pointer arrow on the screen and if you have a mouse attached, it lets you use the mouse to control this little arrow, and you then use this to point to items on the screen and click on the buttons of the mouse when in the right place to tell the computer what to do.
QL mice usually have 2 buttons. Pressing the left button is known as a HIT. Pressing the right button is known as a DO.
There are keyboard equivalents too. The pointer can be moved with the cursor arrow keys on the keyboard. A HIT is (nearly) the same as pressing the SPACE bar key, while a DO is the same as pressing the ENTER key. Remember these key presses if you have no mouse.
WMAN is an abbreviation for the Window Manager. Nothing to do with double glazing salesmen who call after the editor has thrown his PC out of the window again ***[that's enough of that - editor]*** but rather a system whereby a 'standardised' appearance is given to the appearance and operation of menus and so on within the pointer environment system.
HOT_REXT is the part which controls the hotkeys, the so-called Hotkey System II. It runs a little program in the QL called the Hotkey Job which handles the hotkeys.
`HOT_GO` is a command which tells the computer to enable the hotkeys. In other words, once a HOT_GO command has been issued, the hotkeys you have programmed will start to work.
There is a related command called `HOT_STOP` which disables the hotkeys you have defined. I am not absolutely certain why this is necessary but it has to do with the fact that the little program which controls the hotkeys prevents some functions like RESPR from working - you cannot easily load any more toolkits or BASIC extensions while the hotkey job is running without running into a 'NOT COMPLETE' error message, so HOT_STOP allows you to temporarily stop the hotkey job. Note that by default on my system at least, the hotkey job is OFF - hotkeys can be defined, but will not work until you have entered the command HOT_GO. It is necessary for you to remember this in case you run into problems later.
One other thing to explain is CTRL C. Hold down the CTRL key on the keyboard, tap the C key then let go of the CTRL key. This is useful as it lets us switch between the programs running at any given time. You can have a load of programs in memory - this selects which one you want to use at that moment by switching between them all in turn.

Professional & Graphical Software

ProWesS

ProWesS is a new user environment for the QL. ProWesS is short for "PROGS Window Manager", but it is much more than that. Apart from a new window manager, it contains all the system extensions from PROGS, and is essential if you want to run programs which need these extensions.
The ProWesS reader is a major part of the package. It is a hypertext document browser. This means that text files which include formatting commands (including pictures) and possibly links to other files can be displayed and read in this program. This is used in ProWesS to read (and possibly print) the manuals, and display the help files. The hypertext documents which are used by the ProWesS reader are in HTML format, the format which is popular on Internet to display World Wide Web pages.
Another important aspect of ProWesS is the possibility to allow programs to automatically install themselves on your system, and to be able to run them without resetting the system. This means that, when you get a new program, all you have to do is insert the disk and indicate "start the program in flp1_", a menu option in the "utilities" button. To install a program, you indicate "install software", and the software can be added to your system. This way, you don't need to know how to write a boot file to use the multi-tasking capabilities of your computer.
ProWesS includes many programming libraries. These include syslib, an interface to the operating system, PROforma, a vector graphics system, allowing rendering both on screen and on paper (via a printer driver). The DATAdesign engine is also part of ProWesS. It is a relational database system with a bonus, as you don't even need a key field. You get a powerful record at a time data manipulation extension to the language you already use. Of course it also includes ProWesS itself, the new resolution independent window manager.

PFlist

Easy to use program to create listings on any printer (especially inkjet and laser). This ProWesS application allows you to indicate the files which have to be printed. Each column contains a footer which can include the filename and filedate. The listings always allow perforation. PFlist can create your listings in two columns and in landscape (or both).

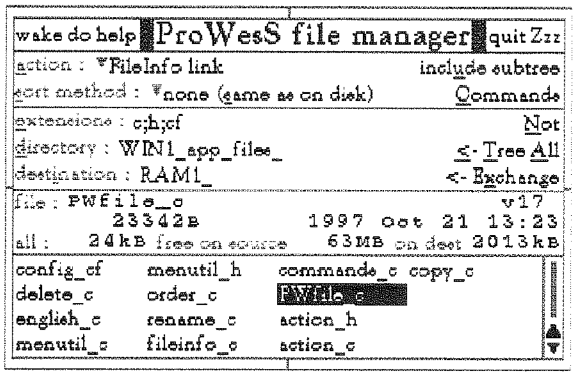
fsearch

File search utility with many useful options, like the choice to search only files with a certain extension, and whether or not the directory tree has to be scanned. All occurences of the searchstring will be displayed with line number or offset. You can also use special matching features, like case dependent, matching a space with a stretch of whitespace, and searching for a word dilimited string.

font-utils

manage your font collection. You can preview fonts on screen, see what characters exist in a font and convert Adobe Type 1 and similar fonts for use in ProWesS.

New ProWesS application
a powerful and very user
friendly file manager



```
wake do help  ProWesS file manager  quit Zzz
action : ▾FileInfo link                  include subtree
sort method : ▾none (same as on disk)           Commands
extensions : c;h;cf                                  Not
directory : WIN1_app_files_                   <- Tree All
destination : RAM1_                           <- Exchange
file : PWfile_c                                      v17
       23342B              1997 oct 21 13:23
all :   24kB free on source    63MB on dest 2013kB
config_cf    menutil_h    commands_c copy_c
delete_c     order_c      PWfile_c
english_c    rename_c     action_h
menutil_c    fileinfo_c   action_c
```

LINEdesign

Create artistic drawings, technical drawings, process bitmaps (even scale and rotate them!), and any kind of vector drawings. You can use grpahics objects to create the most fabulous drawings ever seen. Because LINEdesign is a vector drawing program, any part of the picture can be moved, scaled, rotated, slanted without any loss of precision or resolution. In LINEdesign, pictures are device independant, meaning that the printout will be the same on any printer (e.g. same size and position).
LINEdesign is good at handling text. You can easily put titles and full paragraphs on the page. All the fonts can be displayed at any size, rotation, etc. All the fonts which are available to ProWesS can be used in LINEdesign.
LINEdesign is a drawing program, but it can also be used by people who are not good at drawing. LINEdesign is a great program for making leaflets, posters, and any kind of printed work. Lots of clipart and extra fonts are available from public domain libraries and BBS's. You can even import Adobe Illustrator files.

DATAdesign

Never before has it been so easy to create, fill in and maintain your personal databases. To start a new file, just type the names of the fields. To add or delete a field, no problem, just do it. To change the name of a field, just indicate it. You can choose which fields are displayed and also which records. You can have a hidden comment for each record, look at the file in tabulated form and transfer data to the scrap or hotkey buffer. Files can be memory based (for speed) or disk based (for safety).

new address !!

Dr. Fr. Hemerijckxlaan 13 /1
2650 Edegem
Belgium

tel : +32 (0)3/ 457 84 88
fax : +32 (0)3/ 458 62 07
email : joachim@club.innet.be
www : http://www.club.innet.be/~year2827

ProWesS - BEF 2400 DATAdesign - BEF 1200 PWfile - BEF 900 PFlist - BEF 600
LINEdesign - BEF 1200 fontutils - BEF 1200 fsearch - BEF 600

Payment terms :

You have to run ProWesS to make LINEdesign, DATAdesign, fsearch, fontutils and PFlist work (even though DATAdesign uses wman).
All our software is normally supplied on high density (HD) disks. However they can be obtained on double density (DD) disks at an extra costs of BEF 10C. To use ProWesS and any of our other packages, you need a system with at least 2MB of memory. You should have a harddisk although a two disk system will also work. The use of SMSQ/E is strongly recommended for optimal use of ProWesS.
If you are VAT registered (specify registration number) or live outside the EEC, the amount to be paid is the total (including postage) divided by 1.21 (no need to pay too much).
Payment can be done by EuroCheque in BEF, or by VISA, EuroCard or MasterCard. Credit card orders can be handled by phone. For credit card, please specify name of card owner, card number and expiry date.

Postage : Costs of postage and packaging have to be added. You can choose the quality. Rate depends on no of programs.

	priority mail			*ordinary mail*		
copies	Belgium	Europe	World	Belgium	Europe	World
one	100	200	240	100	120	145
two	135	340	420	135	190	230
3 or 4	160	560	770	160	310	395
5 to 8	185	870	1250	185	550	705
more	295	1130	1610	295	800	1030

All prices are in BEF, including 21% VAT

Now back to the definition above. It is best explained with an example. Suppose you are typing something into a wordprocessor and want to check something in a database program without having to quit from the word processor. On old style computers you'd have to save your file, quit from the word processor, start the database, check the information, quit from the database and start up the word processor again.
But this is the QL. It can multitask and task switch.

Suppose we defined a particular keypress in such a way that it caused another program to start without having to stop the first one. Or if the second program had been started already, that the keypress allowed us to 'jump' straight into the other program which was still in memory. Another key combination we defined is then pressed which takes us back into the first program which had also remained in memory.
This all sounds horrendously complicated to take in at first, but you quickly get used to it after seeing it in action.
This is the basic principle behind hotkeys. We define keys to perform actions. Whatever we were doing, these keys allow us to instantly do something else without having to go through the hassle of quitting, going to basic, entering long winded commands and so on. And the beauty of them is that once you have set up the keypress for a certain action, it can be added to the BOOT program you normally use to set up your computer each time you switch it on.

OK, I have referred to 'actions'. what exactly are these? They can be:
- Loading a program into memory for fast starting
- Loading a program from disk
- Picking a program - jumping straight into it from another program
- Stuffing text or whatever into a special buffer (like a sort of clipboard) to pass it into another program
- Send commands to SuperBASIC
- List which keys are defined
- Enable or disable an individual key definition
- Special features to help with 'difficult' programs like Quill

We'll explore these more as we go further into the series.

Please note: QPAC2 users: there are some extensions which seem to be in QPAC2 rather than hotkeys as such. Buttons for example. I'll stick to what I think is strictly hotkeys for now at least.

Hotkeys are defined as a keypress used in conjunction with the ALT key, e.g. hold down ALT and tap the A key. If you have ever used ALTKEYs from Toolkit 2, you may be wondering what the difference is.
Suppose we defined ALT a to LOAD a SuperBASIC program called UTILITY_BAS:

```
ALTKEY 'a','LOAD FLP1_UTILITY_BAS'
```

We are in SuperBASIC, and we press ALT a (hold down ALT, tap a). Sure enough, the command LOAD FLP1_UTILITY_BAS appears in the channel #0 area at the bottom of the screen. Suppose we are in Quill. Oops, now what was intended to be a BASIC command has now gone into Quill by mistake.
The hotkey system allows us to go one better than this. We can tell the computer that this is a command intended for SuperBASIC, so defining ALT a using a hotkey function instead (HOT_CMD - stands for Hot Command) would ensure that the computer jumps into BASIC first then types in the command for you. The process of jumping into BASIC or a particular program is called PICKING. Think of it as PICKING which program to jump into.

So we can see that the hotkey system gives us more versatility and power than your average simple ALTKEY definition. In fact, both can be freely mixed as long as you remember which is which (a problem for me sometimes).
There is one very special hotkey - the last line recall. ALT ENTER (hold down ALT and tap the ENTER key) 'remembers' the last thing you typed before you last pressed ENTER and types it up for you again. It doesn't always work perfectly, not because of any faults it's just the way it happens to work, but it's very useful when you have to do repetitive typing work for example. Or if you made a mistake entering a command the first time, you can have a second go by pressing ALT ENTER and editing out the mistake.
You can generally define most of the QL keys as hotkeys, but there are a few which are reserved and special and you should avoid trying to redefine them.
Letter and number keys are usually OK unless other programs use them.
Function keys can be defined, but I tend to avoid them hard to remember and prone to mistakes.
ALT ENTER should not be defined because of the last line recall.
The space bar should not be defined as it is used for a special purpose, the stuffer buffer, which we will look at later.
Avoid trying to use the other keys like CAPS LOCK and TAB, although the symbol keys like \, apostrophe and so on are usually OK.

How to define Keys and Actions

The commands to define hotkeys are included as SuperBASC extensions to make them easy to use. Most are FUNCTIONS rather than commands as such. Entering them means they return a value which is a code to tell you if the definition was successful or not.

Try this:

`PRINT HOT_CMD('a','LOAD flp1_UTILITY_bas')`

It prints up the number 0 if it succeeded in defining that command to that key. If it failed - for example, you had already defined something once and not cancelled it before trying to redefine it, some of the functions will return the code for an 'In Use' error (-9)

The thoughtful Mr Tebby provided us with a little command to handle these numbers passed back by the definitions, called ERT, which stands for Error Report. If all went well, it does nothing. If something went wrong it complains by grunting from the computer speaker or gives an error message. Here is an example of how to use it.

`ERT HOT_CMD('a','LOAD FLP1_UTILITY_BAS')`

The first part of the definition in the brackets is the KEY to which the command is to be assigned. Hold down ALT and tap that key to use it. But you should be careful of using upper or lower case keys, as there is a fairly complex rule as to whether upper and lower case keys have the same defintion or not. If you defined a key using a lower case key and no definition exists for the upper case key, both will produce the same result, i.e. ALT a and SHIFT ALT a or ALT A will be the same. You can define a separate action for the upper case key, a way of getting two definitions onto the same key. This can be confusing apart from one special case.

I follow a rule of thumb I read somewhere about a useful guideline for which case of key definition to use for which action.

For actions that involve loading or picking programs:

- use the UPPER CASE key definition to load the program
- use the lower case key definition to pick it

The simple reasoning being that of the two, loading is the less common action, so SHIFT ALT A might be used to load a program, but once it's in memory and you're merrily switching between programs until the computer is sick of you, ALT a is less work because it involves less keys to press. Where you may come unstuck of course is when the CAPS LOCK key is on. But that's your problem.

OK, how to load and pick programs:

There are several ways to load a program. You can put it into memory and start it (or several copies of it!) running from memory to save having to insert its floppy disk each time you need to start it. This is very useful for commonly used programs, but it does tie up a little bit of memory. The simplest one to understand is the command which just loads a program from floppy disk. I'll assume my program is called PROGRAM_EXE in this example:

`ERT HOT_LOAD('p','flp1_program_exe')`

When you hold down ALT and tap p the computer tries to load PROGRAM_EXE from FLP1_. Simple. But again there are a couple of minor points to note.

I have put the text in quotes above. In most case, it could have been written as:

`ERT HOT_LOAD(p,flp1_program_exe)`

If in doubt, use the quotes. Also, the filename of the program was typed in lower case, it doesn't matter, the QL handles upper and lower case filenames! Note that if you omit the drive name, hotkeys will try to add the DATA_USE or PROG_USE default drive name, the Toolkit 2 default drives.

There are two extensions to load programs into memory so that they can be quickly started without having to have their disks in the drives - this is rather similar in principle to using RESPR or LRESPR to load BASIC extensions or Toolkits into memory.

The two extensions are: `HOT_RES` and `HOT_CHP`. Now things get a little more complex! In fact, there are other variations called `HOT_RES1`, `HOT_CHP1` and `HOT_LOAD1` for special cases where you dare not try to execute two copies of the same program at the same time because they are naughty and modify themselves as they run or other such problems. We'll ignore those for now, but bear in mind that they exist and the broad reason for their existence.

HOT_RES is used to load programs RESIDENT. That means they are loaded into memory so that you can quickly start one or more copies of them. If there are other programs running, or the hotkey job is active, this cannot work, so you have to use HOT_CHP instead. In some versions of the hotkey system, the HOT_RES function seems to convert itself into a HOT_CHP automatically when this is a problem.

When should you use HOT_RES or HOT_CHP rather than HOT_LOAD?

Easy. Reserve use of HOT_RES or HOT_CHP to those programs you use most often, to avoid having to insert their floppy disks every time you start a copy. HOT_LOAD is fine for occasionally used programs where you don't want them to hog some memory all the time while they are resident.

The actual sequence of events when a HOT_RES or HOT_CHP command is issued are as follows:

1. When the computer interprets the command, it adds the name of the program to be HOT_xxx'ed to the hotkey system. That means that in essence, when the line defining the hotkey is executed, the program is searched for and loaded into memory.
2. When you later press the hotkey defined as the HOT_CHP or HOT_RES it looks for the program in memory and tries to start it running.

HOT_LOAD is essentially only the equivalent of step (2) - it only loads the program named when you press the keys indicated, rather than storing a copy in memory. It is important to understand the difference between these commands. HOT_LOAD will start a program from disk only when you tell it to. HOT_CHP and HOT_RES will immediately make a copy of the program from floppy disk into memory when they key is defined. Later, when you press the key defined by HOT_RES or HOT_CHP, a copy of the program already loaded into memory is executed. In fact it's rather clever - if you used HOT_CHP or HOT_RES, only one copy of the program is kept in memory even if you have two of those programs running. It gets rather complex to explain, but basically programs defined as 'pure' can have more than one copy of themselves running at the same time, although there is in fact only one copy in memory - the system is being rather clever in running more than one copy of the same code from just one copy of it!

This includes Quill, Archive, Abacus, Easel and the QPAC1 programs for example. But some programs modify their own code in such a way that it is not wise to make one program in memory actually run as though it were 2 separate programs in case one copy modified its own code at the same time as the other incarnation of itself tried to do something completely different and chaos ensues! Turbo compiled programs fall into this category - you have to use the HOT_CHP1 and HOT_RES1 versions of these commands to make sure that there are the same number of physical copies of the program in memory as there are copies of it actually running.

This is not easy to explain, I hope you can follow it. HOT_LOAD does not make that distinction. It always loads a copy off disk and runs that when you need it. If you have problems understanding HOT_CHP and HOT_RES and their variations, ignore them, use HOT_LOAD alone for now until you get more used to things.

Once the program is in memory, you need a way to jump to and from it. Let us assume that our little example program called PROGRAM_EXE has a name of simply "PROGRAM" (I mean the name it displays when it's running and you issue the command JOBS from BASIC). There is a function called `HOT_PICK` whose function is to find the program named, and bring it to be the current job. This action is called PICKING.

```
ERT HOT_PICK('a','PROGRAM')
```

Now when you press ALT a the system looks for a program of that name and brings it up for you to type into it.

This is all well and good, but as usual with hotkeys, the story is not that simple I'm afraid.

Job names are not simple, and may not be obvious, so the hotkey system provides another option concerning job names.

To pick a job or even remove it, you may need to know its name. It's usually held near the beginning of a program (VIEW a program to see it) but you don't have to know how to discover a program's name; worse, some programs change their own names while they run for reasons which are beyond me!

So you can use the next option in some hotkey commands to set the names! This takes a form with 3 parameters:

```
ERTHOT_xxx('key','filename','programname')
```

HOT_xxx can of course be HOT_RES, HOT_CHP etc.

The filename is the same as in the examples above. The program name is any reasonable name you care to give the application. For example, if the program was a database application, you could call it 'mydbase', so the command would then become:

```
ERT HOT_LOAD('D', 'FLP1_DATABASE_EXE',
'mydbase')
```

Once the program is running, you'd see the name as 'mydbase' if you issued a JOBS command or used QPAC2's jobs menu. (A Job is just simply a machine code program, one you can start with EXEC for example).

From then on, we can define a hotkey to pick that program for us, using our upper case/lower case rule above:

```
ERT HOT_PICK('d','mydbase')
```

So that when you press ALT d the system looks for a program which has the name 'mydbase', hopefully finds it and brings it up for your attention.

Right, my brain is now hurting, never mind yours. I suspect this is going to be enough for this issue. Next time, I'll have a look at some extra options to handle naughty programs like Quill.

RWAP SOFTWARE

ON THE MOVE

SBASIC / SuperBASIC Reference Manual (Release 2 - p&p extra) £40

Q-Help v1.03 £10

QL Genealogist v3.23 £25

Genealogy for Windows v5 £55

Flashback SE v2.03 upgrade £5

3D Terrain v1.24 £8

Quick Mandelbrot III v3.01 £5

Sidewriter v1.08 £10

SToQL v1.29 £5

Q-ROUTE MAPS

Britain Map v1.07 (28/4/99) £2

BIG Britain Map v2.01 (27/4/99) £5

West Yorkshire Map v1.01 (28/4/99) £1

Catalonia Map v1.01 (23/6/99) £2

Belgium Map v1.01 (4/2/99) £2

Ireland Map v1.99 (1/12/99) £5

WARGAMES

War In The East MKII v1.24 upgrade £10

D-DAY MKII v3.04 £10

Grey Wolf v1.8 £8

UPDATE YOUR REFERENCE MANUAL:

RELEASE 1 & 2 UPDATES £6 each

ANY 2 UPDATES £10

GAMES

Open Golf v5.20 £8

QuizMaster II v2.07 £5

Stone Raider II v2.00 £5

Hoverzone v1.2 £5

Deathstrike v1.5 £5

ADVENTURES

Return To Eden v3.08 £10

Nemesis MKII v2.02 £8

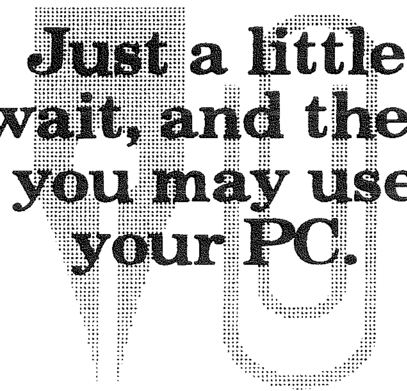
The Prawn v2.01 £8

Horrorday v3.1 £8

West v2.00 £5

The Lost Kingdom of Zkul v2.01 £5

RWAP Software, 4 Anvil Crescent, Coseley, Bilston, West Midlands WV14 8GA
E-MAIL: rwap@qbranch.demon.co.uk
WEB-SITE: http://rwap.webjump.com



Just a little wait, and then you may use your PC.

At JUST WORDS! we are not paranoid about the PC. It is there and many of us have to use it daily. We are therefore writing the most versatile possible utility to transfer files between your QL and PC word processors. Your QL word processor can be Quill, Perfection or Text87, and your PC word processor either Windows or Dos based. The features of the program include:

Correct transfer of accented and similar characters (of special interest to non-english language QL users).

Possible transfer of bold, underlined and italic text (depending on your word processing software).

Setting font, font size, justification, margins and tabs on the QL for windows based word processors.

Versatility and compatibility take time, but we hope this program will be ready by the next issue of QL-Today. Meanwhile our other programs are available at the reduced price of £10 each or £25 the set direct from JUST WORDS! or from QBRANCH.

Geoff Wicks, 28 Ravensdale, Basildon, Essex SS16 5HU, United Kingdom.

Telephone: +44 (0)1268 - 281826

e-mail: geoffwicks@hotmail.com

You and Your Software - Just good Friends?

Part 4 - Problems not solutions.

Geoff Wicks

If Margaret Thatcher were a QL user, she would not be pleased with me. She once famously said of her ministers that only one brought her solutions. The others came with problems. I am coming with problems not solutions.

There was a time when the QL world was much simpler than it is today. If you were not using mdv1_, then you were using flp1_. There were a few non-conformists using fdv1_ or fdk1_, and even brave pioneers using win1_ or hdk1_, but we all had one thing in common. We used a device of three letters, followed by a device number and an underscore. Things changed with the introduction of sub-directories just under 10 years ago. Now there are even people who want device and file names of over 35 letters. 10 years ago you could hack a program to change "mdv" to "flp", but you can no longer do that. 10 years ago you could send information from a boot program to an executable program by poking the details into screen memory, but with high resolution screens you can no longer do that.

If your program has any ancillary files, you need to tell it where to look for those files. There are several ways of doing this, but no agreement among QL users about the best way. Whatever you do you will upset some of your users. I can provide no answer to this, and can only raise the issues.

In its early days JUST WORDS! tried to be all things to all men, and ended up pleasing few. It offered more than one way of telling a program where to look for ancillary files, but the first way interfered with the second way and neither worked satisfactorily. As a contributor to QL Today put it, I had "got into a terrible scramble". However his solution of parameters passed after the file name would mean starting Style-Check 3 with a command like

```
EX   win1_style_style3_obj;'win1_style_,
-1,0,9600,"Bold","par",7,0,2,1,0,40,25,
3,3,3,1,1,1,1,1,1'.
```

Not exactly user friendly.

There are many ways to pass parameters to a program, including DATA_USE, DEV_USE and SUB. Some of these are designed for use with specific hardware or for older software and are best avoided in your programs. It is also possible to pass parameters in an EX (Toolkit 2) or EXECUTE (Turbo Toolkit) command. A configuration block can also be used, but these are controversial. Some users swear by them, but others see them as an antediluvian PC solution to the problem. Another disadvantage is that configuration blocks cannot be incorporated into Turbo compiled programs and instead QLiberator must be used.

In the end JUST WORDS! has opted for a configuration block that can be overridden by a parameter passed in a EX command. This seems to please most customers, but not all. JUST WORDS! configuration blocks are simple and just contain information to point the program to a small definition file containing the program parameters. Some clients and reviewers maintain all the definition file information should be included in the configuration block.

STYLE-CHECK 3 has 21 user-configurable items and it would be a formidable configuration block that included all these items. I hate configuring a program via another program, particularly when things like program colours are involved. I much prefer to see the effects of any changes and to test them straight away.

My definition files can be modified and saved from within my programs, but some users prefer to do this in a text editor. They criticise the lack of information over what is included in the definition file.

I have a favourable impression of QL-ers as customers, but it is an idiosyncratic community, particularly on hard disk use, boot programs and program installation. As a software author you need to be aware of this and that your idiosyncrasies may not be those of your users. In this series I include friendly software rules, but this time it is almost a homily.

FRIENDLY SOFTWARE RULE: You don't have to agree with your critics, but always listen to them.

At shows I often explain to a client why something is impossible or why I am not prepared to make changes to a program. In most cases I have a good reason, but occasionally, when thinking about it later, I realise the client has a point and it might be possible to make some of the changes he has suggested.

Finally let me be a little controversial. There is a group of QL users who have extensive experience of using PCs or other computer systems. Some expect the QL to be like a PC. They cannot envisage a computer that is not constantly accessing a hard disk. Often they want the QL's ram disk to take over the task of a hard disk.

Some users of Solvit-Plus criticise the program saying it would work faster from ram disk. There are even users who copy the files to ram disk before loading, which means that they have two copies of the program in their machine after loading. They then wonder why they are running out of memory. Solvit-Plus, like most well written QL programs, does all its work in memory and uses no temporary files. The use of temporary files, even on ram disk, would slow down the program, not speed it up.

A strong point of the QL is the ease with which you can set up a ram disk, but a consequence is that ram disks are incorrectly and unnecessarily used by some programmers. Even worse when you leave their program, you discover their temporary files are still in the ram disk. If you must use ram disk for temporary files, make sure all the files are deleted by your program quit routine. Similarly if you work within memory, make sure that on quitting all memory used by the program is released. It is the QL equivalent of taking your litter home.

FRIENDLY SOFTWARE RULE: Make sure your quit routine releases all the memory your program has used.

NEXT TIME: More problems without solutions.

■

The Paris International QL Show

Wolfgang Lenerz

On September 18, QLCF, the French QL CLub, organized an international meeting, whose venue was at the Ecole Supérieure d'Informatique (ESI) in Paris, which put a large room at the disposal of the QLers.

It was unfortunate that QLCF, the organizers, hadn't foreseen any kind of refreshments, forcing the participants to skip away from time to time to get something to eat or drink. Well, at least they got a small view of a typical Paris street that way, instead of remaining confined in the room. The room had an impressive security system consisting of an access control through card readers, which, as the door was left open, resulted in a periodic protest by the security system.

The show started at 10 in the morning, and everything was over by about 6 in the evening. This was an international meeting, and there were some international visitors, mainly from Switzerland and Britain. There were quite a few French visitors though, even some of the students of the ESI. They were polite enough not to say that they had come to visit the living dinosaurs... In my personal view, attendance was so-so, however, both Jochen Merz and Tony Firshman told me that it was a good meeting. You will have gathered from this that the usual band of traders attended the meeting, led by Jochen Merz, Tony Firshman and Roy Wood. As usual, Jochen Merz was kept quite busy updating disks whilst Tony Firshman played with his Lego (and, incidentally, repaired a heap of QLs).

Perhaps the most interesting show event was Tony Tebby showing the latest version of the colour drivers for SMSQ/E (for the time being, for the Q40). The bad news is that the drivers were not quite finished - the good news is that they should be finished by the time you are reading this! After the drivers for the Q40, Tony will make those for the QXL... During the show Tony showed a Q40 with working 24 bit colour drivers at a resolution of 1024x512, where things such as QD worked without any problem. "Old" programs, i.e. those expecting a normal mode 4 colour screen worked nicely alongside newer programs, such as a small basic program that Tony wrote during the show (!) to display most of the 24 bit colours.

The bits still missing during the show were really secondary: small things (such as a sprite drawn right under the pointer) were still to be corrected. As mentioned above by the time you read this, the Q40 colour drivers should be ready.

Another important development was that Jonathan Dent showed part of his (as yet unfinished) code through which he hopes to implement a TCP/IP stack for the QL. This will be of great news to any communication happy QLer, since Jonathan plans to implement some form of Email on the QL with this (POP 3 to be precise). There is, of course, no date given for when this will be finished, let's just hope that Jonathan can pull this through!

On Jochen Merz' stand, the news (already a few weeks old at the time, i.e. ages in this information age) was that one could now buy QPC2, which, at DM 80 for users upgrading from QPC1, was nicely priced. For those not yet in the know, QPC2 is the QL Emulator that runs under windows. You'll probably read quite a bit about this somewhere else in these pages, so there is no need to talk much about it, except that - it works: This article was written (in part during the show itself) on Xchange running in QPC2.

As is usual during these rather informal shows, much help was issued to those asking for it - one of the nicer points of the (ever diminishing) QL world is that the dividing line between traders and other users is often quite muddy, and all of the traders are always very helpful, even to those to whom they've never sold anything.

All in all, it seemed a nice meeting, the organisers were even pressed to organised a new one next year...

QL Forever!

QLTdis - part two

Norman Dunbar

In the first instalment of QLTdis, I left you with a functioning program. It functioned, but did nothing of much use that is. In this episode, I am going to discuss the various instruction types that I have spent many hours decoding. The following describes each one and follows roughly in order of complexity.

This infomation is the heart of the disassembler as it defines how we decide which instruction we have found and what type it is.

The type decides which decoding sub-routine we will have to use to completely disassemble the instruction.

Type zero.

This is the simplist instruction of all. It consists of a single word and has no variable bits that define additional information. There are seven type zero Instructions and the mask for type zero is $FFFF. The instructions, with the result of ANDing the mask are:

ILLEGAL	$4AFC
RESET	$4E70
NOP	$4E71
RTE	$4E73
RTS	$4E75
TRAPV	$4E76
RTR	$4E77

Type one.

This is a similar instruction family to those above, however, there is a data word following the instruction word. In other words, this is a 4 byte instruction. The first word defines the actual instruction and bits 0 to 7 of the following word (ie byte 4 of the entire instruction) defines the data. There are three type one instructions, these are, along with the result of ANDing with the mask word - $FFFF, as follows:

ANDI #data,CCR	$023C
EORI #data,CCR	$0A3C
ORI #data,CCR	$003C

Type two.

This is identical to type one, except that all 16 bits of the data word are used. Once again, the mask word is $FFFF and three of the four type two instructions act upon the SR (Status register) and are:

ANDI #data,SR	$027C
EORI #data,SR	$0A7C
ORI #data,SR	$007C
STOP #data	$4E72

Type three.

There are two type three instructions. The instruction is two bytes long but bits 0 to 2 define the number of an address or data register (0 to 7). For this instruction family we use a mask word of $FFF8 and the instructions are:

UNLK An	$4E58
SWAP Dn	$4840

Type four.

The type four instruction - there is only one - was outlined in the introduction to QLTdis, however, to recap, it is the TRAP #data instruction and this has the data in bits 0 to 3 of the two byte instruction word. This requires a mask of $FFF0 and the result is:

TRAP #data	$4E40

Type five.

The is only a single type five instruction. This family has an address register number in bits 0 to 2 - similar to type three - but also has a word of data following. This is a 4 byte instruction. The mask is once again $FFF8 and the instruction and its result is:

LINK An,#data	$4E50

Type six.

Type six instructions are the 'DBcc Dn,displacement' family. This has 4 bytes in the full instruction and bits 0 to 2 define the register number while bits 8 to 11 define the condition code (cc). The second word is used to specify the displacement for the branch. The mask is therefore $F0F8 and the result will be $50C8 for all cases. Once we have calculated this, we can send 'DB' to the buffer and extract a condition from the value in bits 8 to 11. The set of values are :

0	T
1	F (also RA)
2	HI
3	LS
4	CC
5	CS
6	NE
7	EQ
8	VC
9	VS
10	PL
11	MI
12	GE
13	LT
14	GT
15	LE

In QLTdis, we will generate a DBF instead of DBRA because this means that we can use a subroutine for the DBcc, Bcc and Scc instructions. Now that we have extracted the condition, we can send it to the buffer, extract the register number and then the displacement word. It is starting to get complicated !

Type seven.

These are the BSR instructions, of which there are two - the short and the long form. The mask is $FF00 because the data is in bits 0 to 7 of the instruction word. This gives:

BSR	$6100

If the displacement byte is zero, then this is the long form and the next word defines the 16 bit displacement. This is the 4 byte BSR instruction. If the displacement byte is not zero, then this is the BSR.S short variant and the instruction is only two bytes long.

Type eight.

This family contains the Bcc and the BRA instructions. Both sets have a long and short form, similar to BSR above. The displacement is defined by bits 0 to 7 and the condition codes are defined by bits 8 to 11. This gives a mask word of $F000 and a result of :

BRA	$6000
Bcc	$6000

As you can see we might just have a problem. The mask gives the same result for both instructions - how can we determine which is which? The condition held in bits 8 to 11 is our rescuer. If this is zero then this is a BRA and if not, it is a Bcc. The condition codes from 1 through 15 are as per type six above. This is where we can use the same routine as we used for DBcc to extract the condition code for our instruction. It is also

why we have to use 'DBF' instead of 'DBRA' for condition value 1. If not, we would get DBRA which is ok, but then we would get BRA for condition code 1 as well - this is wrong because BRA is condition code 0!
Once again, if the displacement in bits 0 to 7 is zero, this is the long format and the following word has the true displacement, otherwise this is a short form.

Type nine.

There is a single type nine instruction in which a register number is defined in bits 0 to 2 and the instruction size is held in bits 6 to 8. The mask must therefore be $FFB8 and the instruction is:

EXT.size Dn	**$4880**

The size will be zero for EXT.W Dn and 1 for EXT.L Dn.
This is a relatively simple instruction. It does have a catch however. Other instructions also specify a size and the size is normally 0 for .B, 1 for .W and 2 for .L this instruction has a different set of values and .B is not allowed.

Type ten.

Again, there is only one instruction in this family. There is a register number in bits 9 to 11 and the data is in bits 0 to 7. This means that the mask will be $F100 giving a result of :

MOVEQ #data,Dn	**$7000**

Type eleven.

Bits 0 to 2 define a source register, bits 9 to 11 the destination register and bit 3 defines whether the instruction is operating on memory or not. This family has a mask word for each member. The instructions, masks and results are:

ABCD	**$F1F0**	**$C100**
(Mask then result)		
SBCD	**$F1F0**	**$8100**
ADDX	**$F130**	**$D100**
SUBX	**$F130**	**$9100**

If bit 3 is set then the address register with predecrement form is used:

ABCD -(Ax),-(Ay)
SBCD -(Ax),-(Ay)
ADDX -(Ax),-(Ay)
SUBX -(Ax),-(Ay)

and if not, the data register form is used:

ABCD Dx,Dy
SBCD Dx,Dy
ADDX Dx,Dy
SUBX Dx,Dy

Quite a tricky instruction.

Type twelve.

A little simpler this one. There is only one instruction here but it has two formats. There is a register number in bits 0 to 2 and a direction in bit 3. We need a mask word of $FFF0 and we get the following results:

MOVE An,USP	**$4E50**
MOVE USP,An	**$4E50**

The value in bit 3 defines which format is to be used. If bit 3 is set, the direction is USP to An and if not set, it is An to USP.

Type thirteen.

Another single instruction family. This one has a mask of $F138 and a result of $B108 shows that we have decoded this:

CMPM (Ax)+,(Ay)+

where Ax is defined in bits 0 to 2 and Ay in bits 9 to 11. Bits 6 and 7 define the size:

0	.B
1	.W
2	.L

As this is the most common value for the size suffix, this will also be extracted to a sub-routine as there are a few instructions which need a size specifier.

Type fourteen.

This is the first of the instruction families where a mode is specified. Don't worry this is just like the size specifier except that is varies in value and meaning over a number of instructions. This family has variable data in bits 0 to 2, 3 to 7 and 9 to 11. There is only a single instruction in this family but it has three different modes :
The mask word is $F100 and the result will be $C100. The source register is defined in bits 0 to 2, the destination register in bits 9 to 11. The mode is in bits 3 to 7 and defines the instruction format to use as follows :

8	**EXG Dx,Dy**
9	**EXG Ax,Ay**
17	**EXG Dx,Ay**

If the 'mixed mode' value is found in bits 3 to 7 then bits 9 to 11 define Dx and bits 0 to 2 define the Ay register.

Type fifteen.

Type fifteen has a mode, a size source and destination registers and the following word defines a displacement ! This is a 4 byte instruction and has a mask of $F008 and gives a result of $0008. There is only one instruction in this family. It is the 'strange' MOVEP instruction.
Bits 0 to 2 define the address register. The size in

bit 6 is set for long and reset for word. There is no byte sized version. Bits 7 and 8 define the mode and bits 9 to 11 define the data register number. There is a single word of data following which defines the data part of the instruction and all 16 bits are used.
The mode bits, in binary, define the method of the MOVEP as follows:

10	**Memory to register mode - MOVEP data(An),Dn**
11	**Register to memory mode - MOVEP Dn,data(An)**

Type sixteen.

This family includes the shift and rotates and is very complicated to decode.
Bits 0 to 2 define the register to be shifted - Dy.
Bits 3 and 4 define the shift/rotate operation as follows (in binary):

00	ASL/ASR
01	LSL/LSR
10	ROXL/ROXR
11	ROL/ROR

Bit 5 defines whether the shift counter is held in a register (bit 5 set) or in immediate data (bit 5 reset).
Bits 6 and 7 define the size and the mode together. In binary this is:

00	**Byte size shift (in register)**
01	**Word sized register shift**
10	**Long sized register shift**
11	**Word sized shift in memory**

Bit 8 is the direction of the shift - if set, direction is left, reset implies shift right.
Bits 9 to 11 hold the register - Dx - if the count is in a register (bit 5 set) or the actual count if immediate data specifies the shift count (bit 5 reset). If the count is zero, then the shift is 8.
If the shift is in memory (size = 3) then bits 0 to 5 define the 'effective address' of the location that is to be shifted. Effective addresses are discussed next as the remainder of the instruction decodings all require an effective address in bits 0 to 5.
This family requires a mask of $F000 and has a result of $E000 for all members. The members are:

ASL Dx,Dy
ASL #data,Dy
ASL ‹ea›
ASR Dx,Dy
ASR #data,Dy
ASR ‹ea›
LSL Dx,Dy
LSL #data,Dy
LSL ‹ea›
LSR Dx,Dy
LSR #data,Dy
LSR ‹ea›
ROL Dx,Dy
ROL #data,Dy
ROL ‹ea›
ROR Dx,Dy
ROR #data,Dy
ROR ‹ea›
ROXL Dx,Dy
ROXL #data,Dy
ROXL ‹ea›
ROXR Dx,Dy
ROXR #data,Dy
ROXR ‹ea›

Effective address decoding.

This is where the nightmare begins. Effective addresses are defined in bits 0 to 5 of all the remaining instructions. Remember all those addressing modes you have learned since part one of this series? This is where you work them out.
The 6 bits used for the effective address are split as follows :
Bits 0 to 2 - register number
Bits 3 to 5 - mode.
The modes, in decimal, are:

0	**Dn or Data register Direct.**
1	**An or Address Register Direct.**
2	**(An) or Address Register Indirect.**
3	**(An)+ or Address Register Indirect with Post-Increment.**
4	**-(An) or Address Register Indirect with Pre-Decrement.**
5	**d(An) or Address Register Indirect with Displacement. (Requires one extra word for the displacement)**
6	**d(An,Xn) or Address Register Indirect with Index. (Requires one extra word for the index register details and the displacement).**

The extra word is made up of the following:
Bits 0 to 7 - define the displacement.
Bits 8 to 10 - always zero.
Bit 11 - Index register (Xn) size, 0 = word, 1 = long
Bits 12 to 14 - define the index register number.
Bit 15 - defines whether the index register is data or address. 0 = Data, 1 = Address.
If the mode is 7 then we have a special situation where the register number is not used as a register number, but as a sub-mode, as follows:
0 = xxx.W or Absolute short address - one word of data is required.
1 = xxx.L or Absolute long address - two words of data are required.
2 = d(PC) or Program Counter with Displacement - one word of data is required for the displacement.

3 = d(PC,Xn) or Program Counter with Index - one word of data is required for the displacement and index details. This is formatted as above for mode 6 instructions.
4 = #xxx or Immediate Data. Requires one or two additional words for the data depending upon the size of the instruction. Byte sized uses the low byte of the following word. Word sized used the following word and long sized instructions use the next two words as data.
The decoding of the effective address will be farmed out to a subroutine as so many instruction families use it. This is probably the most complicated decoding that we will have to do, but we can do it! Back to the remaining instruction families.

Type seventeen.

The first of the instructions with an effective address. But the simplist as the only variable bits are those in bits 0 to 5 which spefcify the effective address. This family therefore has a mask word of $FFC0 and the following results:

MOVE SR, ‹ea›	$40C0
MOVE ‹ea› ,CCR	$44C0
MOVE ‹ea› ,SR	$46C0
NBCD ‹ea›	$4800
PEA ‹ea›	$4840
TAS ‹ea›	$4AC0
JMP ‹ea›	$4EC0
JSR ‹ea›	$4E80

Type eighteen.

Bits 0 to 5 as above, specify the effective address and bits 6 to 7 define the size of the instruction. The mask word is $FF00 and the results will be:

ORI #d, ‹ea›	$0000
ANDI #d, ‹ea›	$0200
SUBI #d, ‹ea›	$0400
ADDI #d, ‹ea›	$0600
CLR ‹ea›	$4200
CMPI #d, ‹ea›	$0C00
EORI #d, ‹ea›	$0A00
NEG ‹ea›	$4400
NEGX ‹ea›	$4000
NOT ‹ea›	$4600
TST ‹ea›	$4A00

Size is:
0 = byte, 1 = word, 2 = long.

Type nineteen.

Bits 0 to 5 specify the effective address, bits 6 to 7 define the size of the instruction and bits 9 to 11 define the actual data. The mask word is $F100 and the results will be:

ADDQ #data, ‹ea›	$5000
SUBQ #data, ‹ea›	$5100

Size is:
0 = byte, 1 = word, 2 = long.

Type twenty.

Type twenty are the Bit Test, Change, Clear and Set instructions. Bits 0 to 5 are the effective address. Bits 6 and 7 define the actual operation as follows:

00	BTST #data, ‹ea›
01	BCHG #data, ‹ea›
10	BCLR #data, ‹ea›
11	BSET #data, ‹ea›

Bits 0 to 7 of the following word define the bit number that is to be set etc. Following this is the data that makes up the effective address. The mask is $FF00 and the result is $0800.

Type twenty-one.

These are the same as type twenty, but this time the bit number to be affected is in a data register. The effective address is in bits 0 to 5 as usual, bits 6 and 7 define the operation and bits 9 to 11 define the data register where the bit number is to be found. The mask is $F100 and the result is $0100. The operations, as defined by bits 6 and 7 are:

00	BTST Dn, ‹ea›
01	BCHG Dn, ‹ea›
10	BCLR Dn, ‹ea›
11	BSET Dn, ‹ea›

Type twenty-two.

This family holds the multiply and divide instructions, and the CHK one as well. Bits 0 to 5 define - guess what? (The effective address !) Bits 9 to 11 define the data register being operated upon. The mask is $F1C0 and the results are:

CHK ‹ea› ,Dn	$4180
DIVS ‹ea› ,Dn	$81C0
DIVU ‹ea› ,Dn	$80C0
MULS ‹ea› ,Dn	$C1C0
MULU ‹ea› ,Dn	$C0C0

Type twenty-three.

The LEA instruction is the only one in this family. The variable bits are 0 to 5 for the effective address and bits 9 to 11 define the address register being used. The mask is therefore $F1C0 and the result is $41C0 for:
LEA ‹ea› ,An

Type twenty-four.

This faminly includes the generic form of the ADD, SUB, OR and AND instructions and the Address register forms of ADD and SUB - ADDA and SUBA. These two introduce a bit of 'twiddling' to decode them. Bits 0 to 5 have their usual value, bits 6 and 7 define the size, bit 8 the direction and bits 9 to 11 the data register involved. The mask is $F000 and the results are:

ADD	$D000
ADDA	$D000
AND	$C000
OR	$8000
SUB	$9000
SUBA	$9000

The direction bit specifies the format of the instruction:

0	ADD ‹ea› ,Dn
1	SUB Dn, ‹ea›

The size is:

0	byte
1	word
2	long
3	ADDA or SUBA

In the case of size being 3, bit 8 actually defines the size as the ADDA and SUBA instructions only have one format. Bit 8 is:

0	ADDA.W ‹ea› ,An (or SUBA.W)
1	ADDA.L ‹ea› ,An (or SUBA.L)

Type twenty-five.

Bits 0 to 5 are the standard effective address bits which we should all be used to by now! Bits 6 and 7 define the size, bit 8 the mode and bits 9 to 11 define the data register. The mask is $F000 and the results is always $B000. The size and mode bits are used to correctly decode the actual instruction. Size is:

0	byte
1	word
2	long
3	CMPA

If size is 0, 1 or 2, then mode defines the instruction:

0	CMP ‹ea› ,Dn
1	EOR Dn, ‹ea›

If size is 3 then the instruction is CMPA and mode defines the size:

0	CMPA.W ‹ea› ,An
1	CMPA.L ‹ea› ,An

Type twenty-six.

This is the Scc instruction. It has a family all to itself. Guess what bits 0 to 5 represent? Bits 8 to 11 define the condition. The mask is $F0C0 and the result is $50C0. The conditions are exactly as for Bcc and DBcc above. (There is no SRA instruction only SF)

Type tewnty-seven.

Bits 0 to 5 are the usual. Bit 6 is the size, bit 10 the direction and there is a single word of data following to define a register list. After that word, there may be others defining the effective address - if required. The mask is $FB80 and the result is $4880. The instructions are:

MOVEM Dn/An, ‹ea›
MOVEM ‹ea› ,Dn/An

The size is:

0	word
1	long

The word following the instruction is a mask that defines a register number. If a bit is set, then that register is in the list. The list is as follows for control mode and post-increment:

bit 15=A7 bit 14=A6 ... bit 8=A0 bit 7=D7 ... bit 0=D0

And for pre-decrement, the mask word is the other way round:

bit 15=D0 bit 14=D1 ... bit 8=D7 bit 7=A0 ... bit 0=A7

Type twenty-eight.

The MOVE ‹ea› ,An instruction is the only one in this family. The variable bits are 0 to 5 - as usual, 9 to 11 define the destination address register and bits 12 and 13 the size. The mask is $C1C0 and the result is $0040. The size is non-standard in this instruction:

0	byte
2	long
3	word

Type twenty-nine.

Nearly done now! This is probably the most difficult instruction to decode as it has two effective addresses in it. There is only one instruction in this family - the MOVE ‹ea›,‹ea›. The mask is $C000 and the result is $0000. Bits 0 to 5 define the effactive address of the source and bits 9 to 11 define the effective address of the destination. Bits 12 and 13 define the size as follows:

2	long
3	word

Type thirty.

This is the catch all at the end of the list. Anything which has not been detected by this time, will always be caught here. The mask is $0000 and the result is $0000. If we catch an instruction here, the chances are that it was data and to show this, we simply define it as a constant word of data. The output is:

DC.W	$xxxx

Finally....

So that is all there is to it. At the time of writing I have spent many hours going through bit patterns for all the instructions, sorting them into similar groups, then working through each group finding masks and results. This is what led to Dilwyn having the cartoon in the last issue of volume 3 where I was in bed dreaming in binary. It was very nearly true !!

At the start of this article I had decided upon thirty three different groups. As I typed it up, I noticed that a few looked like they could be combined and, after a bit more studying of the reference manual, I ended up with twenty nine, plus the catch all at the end. I suspect George Gwilt could probably combine a few more - he is a much better programmer than I am.

More on QLTdis soon. The next article will be back to the tutorial and covers Exceptions and how to handle them. See you then.

Programmer's choice; using IF and SELect - 1

Mark Knight

:Programmers often need to write programs that must make choices; an editor program may add a character to the text if one set of keys is typed at the keyboard but call up menus if others are typed; a spreadsheet must do one set of calculations if it finds one formula character, another if it finds a different one; the list is endless. Just as it contains two looping structures, FOR and REPeat, SuperBASIC also contains two structures for making choices, IF and SELect. Used well these structures can produce fast, efficient programs that are also clear to read; used badly they will produce a confusing mess that may or may not work, but it will certainly be harder to read and therefore harder to debug.

The first discussed is the IF construction, an essentially simple structure; the powerful SELect structure takes more space, as it has more variations and subtleties, as well as more pitfalls. Once again a few Minerva and SBASIC additions are described along with Turbo and Q-Liberator extras. Most of the time the code has been tested on a JS ROM QL and sometimes on SBASIC; usually it makes little difference, the examples given are intended to be portable between interpreter versions and all should work with both Turbo and Q-Liberator.

IF only; the SuperBASIC IF

To begin with long form IF structures are discussed, followed by the short form with some warnings. Most programming languages contain some sort of IF structure; The SuperBASIC IF has the general forms:

```
IF <condition> THEN
 REMark Do something.
END IF
```

...or:

```
IF <condition> THEN
  REMark Do something.
ELSE
  REMark Do something else.
END IF
```

Note; although THEN is not needed, I prefer to use it; when scanning through a listing it serves as a visible flag to help the eye and brain find any IFs more easily. A simple example might be:

```
100 FOR Test=1 TO 10
110   IF Test=5 THEN
120     PRINT "Test=5"
130   END IF
140 END FOR Test
```

...or:

```
100 REPeat Test
110   INPUT#0;Test$
120   IF Test$="" THEN
130     EXIT Test
140   END IF
150   PRINT Test$
160 END REPeat Test
```

Using ELSE provides more flexibility, as follows:

```
100 FOR Test=1 TO 10
110   IF Test=5 THEN
120     PRINT "Test=5"
130   ELSE
140     PRINT "Test<>5"
150   END IF
160 END FOR Test
```

```
100 REPeat Test
110   INPUT#0;Test$
120   IF Test$="" THEN
130     EXIT Test
140   ELSE
150     PRINT Test$
160   END IF
170 END REPeat Test
```

Most readers won't need to type these in and run them to know what would happen, but to really appreciate how IF evaluates conditions we need to delve deeper. What exactly does SuperBASIC use to decide if the condition described is met? The condition is in fact treated like a single expression that evaluates to zero or non-zero, and zero is false while anything else is true. To clarify this, look at this example:

```
100 CLS
110 IF 0 THEN
120   PRINT "Something is very wrong with this
    computer."
130 ELSE
140   PRINT "Whew, that's a relief!"
150 END IF
```

Hopefully you'll get the second message if you run it. Once this is done, look at this:

```
100 CLS
110 REPeat Testing
120   SomeNumber=RND(0 TO 15)
130   IF SomeNumber THEN
140     PRINT "Non zero, in fact ";SomeNumber
150   ELSE
160     PRINT "Zero!"
170     EXIT Testing
180   END IF
190 END REPeat Testing
200 BEEP 10000,10
```

...this program will work exactly the same whether you use it as listed or change line 130 to:

```
130 IF SomeNumber<>0 THEN
```

The "<>" means "not equal to" (or if you want to be really pedantic it means "less than or greater than" as the "<" is the "less than" symbol and ">" is the "greater than" symbol).
It may seem that multiple conditions can be built using AND and OR, but this is simply a compound condition and as far as SuperBASIC is concerned it is still a single expression with a single value. To illustrate this we look at the following:

```
100 CLS
110 AnyOldRubbish=1
120 SomethingElse=2
130 IF AnyOldRubbish=2 OR SomethingElse=2 THEN
    BEEP 10000,10
```

...surely this is checking two conditions not one? No, it's checking a single compound condition: to prove it, alter the program to:

```
100 CLS
110 AnyOldRubbish=1
120 SomethingElse=1
130 PRINT AnyOldRubbish=2 OR SomethingElse=2
```

...run the program, then change line 120 back to:

```
120 SomethingElse=2
```

...and run it again. In both cases we obtain a single number as line 130 is reached, 0 if the condition is not met and 1 when it is. Even the most complex compound conditions are evaluated as a single number, for example this one:

```
100 BORDER 1,2
110 CLS
120 PumpOn=RND(0 TO 5)
130 ValveOpen=RND(0 TO 5)
140 OilDepth=RND(6 TO 20)
150 SafeLevel=RND(17 TO 25)
160 IF PumpOn AND ValveOpen OR OilDepth>8 OR
  OilDepth<SafeLevel THEN
170   PRINT "Yes, it will work."
180 ELSE
190   PRINT "No, it won't work."
200 END IF
```

To prove it still has a single value, try this version:

```
100 BORDER 1,2
110 CLS
120 PumpOn=RND(0 TO 5)
130 ValveOpen=RND(0 TO 5)
140 OilDepth=RND(6 TO 20)
150 SafeLevel=RND(17 TO 25)
160 PRINT PumpOn AND ValveOpen OR OilDepth>8
  OR OilDepth>SafeLevel
```

...this still prints a single number on line 160, varying between 0 and 1 depending upon the values of the test variables used. When using variables alone in an IF we can simply use code like this:

```
100 CLS
110 AnyOldNumber=1.1
120 IF AnyOldNumber THEN PRINT "True.":ELSE
    PRINT "False."
```

...and insert various values into line 110 of this tiny program to see what happens. What you will find if you try it is that zero is false and anything else is treated as true. IF uses a floating point value and if an integer variable is used it is first converted, so it is not faster to use an integer in cases like the above. However, we can make a suitable condition using an integer, so instead of:

```
IF AnyInt% THEN
```

...use the following:

```
IF AnyInt%<>0 THEN
```

...which is faster since the variable AnyInt% does not have to be converted into floating point and the expression evaluator can quickly stack a ready made 1 as the value of the expression. This applies to the interpreter as well as to Turbo compiled code, though it will make programs slightly larger as well as faster. So IF works with integer values but is slightly faster using its natural floating point form because it avoids conversions.

What the IF structure is looking for is really a single true or false value, based upon a floating point number. If the condition evaluates to true (in other words non-zero) then the code within the IF structure is carried out: if the condition evaluates to false (in other words zero) then if there is an

ELSE included then the code within that structure is carried out. The next job in either case is to look for the END IF and carry on with any program statements after that.
The short form of IF is handy for very simple tasks, like this program fragment:

```
10300 IF Displaying%=1 THEN AT 0,0 : PRINT
      "Valid items found=";Valid;
```

If there are statements after the THEN (or the condition if THEN is left out) then no END IF is needed; the interpreter works out that the IF structure ends with the current line. If a long multi statement line would otherwise be required it is usually better to use the long form of IF since it is easier to read and to update should your program need it. Instead of the above we could have used:

```
10300 IF Displaying%=1 THEN
10310   AT 0,0
10320   PRINT "Valid items found=";Valid;
10330 END IF
```

This doesn't bring much benefit in this case, but if there were five or six program statements it would be very much easier to read the long form IF structure; in addition it is easier to add further program statements if they are needed in the long form. All this certainly should not discourage you from using single line IF clauses in your programs, but use them with thought, not just to save typing.
Another reason to avoid some uses of short form IF is that they may not work properly because of bugs in the SuperBASIC interpreter; this applies to Minerva and probably to SBASIC too. Lines like this:

```
11200 IF Loaded=1 THEN PRINT FileName$:ELSE
      :PRINT "No file":END IF :TestFile
```

...will run into bugs in Minerva 1.97 and the PROCedure called TestFile may be called more than once in this example. One solution is:

```
11200 IF Loaded=1 THEN PRINT FileName$:ELSE
      :PRINT "No file"
11210 TestFile
```

...or better still in some cases:

```
11200 IF Loaded=1 THEN
11210   PRINT FileName$
11220 ELSE
11230   PRINT "No file"
11240 END IF
11250 TestFile
```

..which is easier to change should you wish to add to it later. Nesting single line IF clauses is asking for trouble, and trouble rarely declines an invitation; the following also runs into trouble on a Minerva 1.97 system:

```
11200 IF Choice=1 THEN EXEC_W "Quill":ELSE IF
      Choice=3:EXEC_W "Abacus":END IF:NEXT
      Loop
```

...and will not work properly (never mind the details!)

Fix: One fix could be:

```
11200 IF Choice=1 THEN EXEC_W "Quill":ELSE :IF
      Choice=2:EXEC_W "Abacus":END IF:NEXT
      Loop
```

...in other words, add a colon after the ELSE! A better fix is don't use these ridiculous nested single line clauses and write code like this:

```
11200   IF Choice=1 THEN
11210     EXEC_W "Quill"
11220   ELSE
11230     IF Choice=2 THEN
11240       EXEC_W "Abacus"
11250     END IF
11260   END IF
11270   NEXT Loop
```

...or better still use SELect for this kind of thing! Although both of these are Minerva problems I would not claim that there are no bugs in the interpretation of IF using other systems, just that these are the bugs I know about.
Avoiding overcomplicated single line IF clauses is a good way to avoid errors of your own not just bugs in the interpreter, and it certainly makes programs easier to read. Remember you may want to alter that program you are writing long after you have forgotten how it works, so writing it in a manner that makes it clear is always a good idea. If you have to come back to a SuperBASIC listing years after it was written you will be glad of meaningful variable names, proper indenting and the avoidance of overlong multiple statement lines.

Next issue: Being SELective: The SuperBASIC SELect structure.
■

SCART Connections - Part 1

Richard Cooke

I found the article on page 54 of the last issue about SCART misleading. Here is a better answer, including calculations.

I used the remote control to switch the TV set into QL mode.

Earth on Scart pins 5,9,13 ought to have been connected, but I'm lazy - Scart pin 17 was earthed.

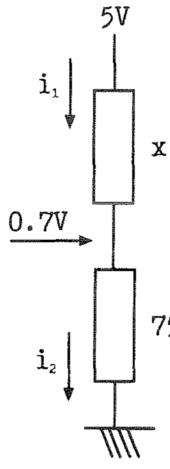
SUMMARY

Buy 4 resistors:
1 1K2 ohms for CSYNC
3 680 ohms for RGB
Then solder them in series with the QL signals inside the Scart plug. The 75 ohm resistors are inside the monitor.

1. QL RGB to SCART

Video outputs from QL RGB are 5Volts peak-to-peak. Attenuate to 0.7volts peak to peak. See diagram in Fig. 1

Fig. 1



Assume current $i_1 = i_2$
Using V=IR it follows that i=V/R
0.7/75 = i = (5-0.7)/x
(75*(5-0.7))/0.7 = x = 460 ohms

A higher value for x will make the screen darker, and reduce the overloading of the IC22 ULA ZX8301.

Preferred values 330, 470 or 680 ohms, the nearest to 460 ohms is 470 ohms, the resistor colour code for which is Yellow, Violet, Brown
Fig 2 shows how this is wired into the SCART connector.

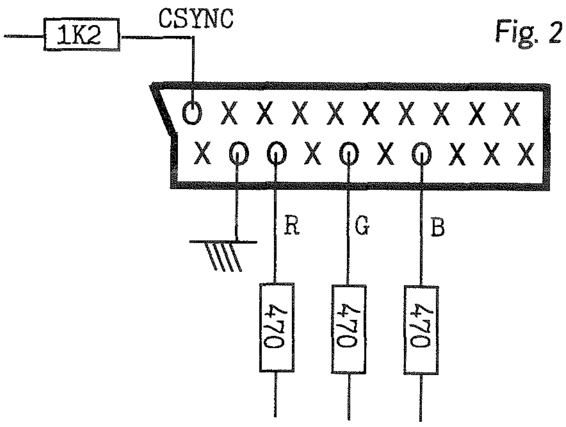


Fig. 2

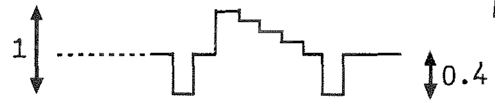
QBOX-USA BBS

Operating since 1993 on a Sinclair QL from Utica, Michigan, USA
Supporting *ALL* Sinclair and Timex users
Message and File Areas for QL, Z88, Spectrum, TS2068, ZX81, TS1000
Modem speeds 300 bps to 33.6k bps supported
24 hour operation - call anytime
810-254-9878

Practical Result

The picture still appeared slightly too bright in comparison to normal television.
Video output from QL RGB signals = 5 volts peak-to-peak. Now see Fig. 3

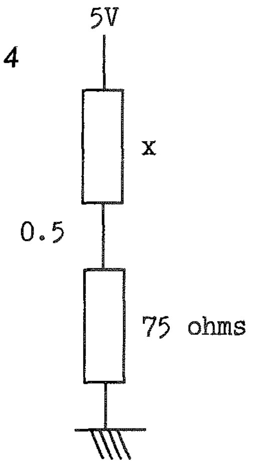
Fig. 3



Monitor wants 0.7Vpp, that's (1 * 0.7)/(1+0.4) = 0.5 volts excluding sync.

Now see Figure 4.

Fig. 4



Find the value of x (the series resistor).
V=IR and V/R = I
0.5/75 = (5-0.5)/x
so x = 675 ohms.
Nearest preferred resistor value is 680 ohms.

Practical Result

Brightness and contrast controls on the monitor did not require adjusting.

Tips

Place 680 ohm resistors to SCART pins 7,11,15. Place heat shrink sleeving over each. Solder RGB wires from the QL to the resistors. Use 5 core cable from QL to SCART. Use Military type high temperature, not Burglar Alarm cable. Heat the heat shrink to insulate.
The military specify high temperature cable because they are no good at soldering like the rest of us. It's nothing to do with them wanting electronic equipment to work after it's been blown up!

Next Issue: QL composite sync to SCART.
■

EPSON Printer Control Codes

Ian Pizer

I have experimented with using EPSON Printer ESC/P2 codes for printing from DataDesign. Most of the ESC/P2 codes are straightforward except for ESC X (m) (nL nH) which allows choice of the point size from up to 32 but the means of using the code is obscure. Here is what I have found thanks to various articles in QUANTA and QL Today:
`open#3,par`
`bput#3,your desired codes` (see below).
`27,88,1,16` will give 8 point characters
`27,88,1,21` for 10 point
`27,88,1,24` for 12 point
By incressing the last number by steps of 4 you can get up to 32 point characters.
Example:

```
open#3,par
bput#3,27,88,1,48
print#3,"Some text to be printed"
```

will print at your printer in 24 point characters.
■

More THINGS... - Part 2

Jochen Merz

In the first part of this series, I (hope that I) gave you an idea what Things are. How can they be accessed, created, how does the system control them? QDOS and Minerva have no operating system calls for adding and removing Things, so a different way had to be found.
Let me start a bit technical and afterwards relax during this article I promise I will keep the technical section as short as possible. Operating system calls like memory allocation, linking in drivers etc. are all grouped under "Trap #1". This is a method of doing an operating system call in machine code. A register (D0) contains the "function code". Thing calls belong into the same group, but as the QL operating system is in ROM, it cannot be modified. Furthermore, the list in ROM cannot be extended, there was no provision for "hooks" etc. When a programmer tries to call a Trap #1 with a function code for Thing calls, the operating system returns "not implemented". (Just for the interested ones: some implementations of

SMS, mainly SMS2, contains the Thing calls as "real" Trap #1 calls). When the programmer tries to perform a Thing call and he gets a "not implemented", then he has to use the "alternative way" of doing it - which should always work. The HOTKEY System II extension does not only add improved HOTKEY facilities to QDOS and Minerva systems, it also adds the Thing calls in a slightly different way. I am not going into details now, but a programmer can easily find the Thing calls and call them with the same parameters and a function code in D0 - and they work in exactly the same way. Now the really technical bit ends.

So, in order to get Things, you need to have the HOTKEY System II loaded. It is built into SMSQ/E but needs to be loaded seperately on Minerva and QDOS. If you would like to see lists of Things on your display then you should also load QPAC2. QPAC2 consists of Things and it allows you to list them, so its useful in two ways.

Type the following instruction into BASIC

```
EXEP "Things"
```

A window will appear showing you a list of Things in your system. And ... you already used a Thing called "Things". It is an executable Thing installed by QPAC2 and provides you with a job which lists the Things. So you can start programs without actually accessing any file. If you are a bit confused here, please read part 1 again - it dates back 4 months so you may have forgotten most of the theory described therein.

If we look at the Things available, you will find Things like "Button", "Exec", "Jobs", "HOTKEY", and, fairly down at the bottom of the alphabetically ordered list, "Things". This is it!! That's what you executed with the EXEP command.

Try something else, for example

```
EXEP "Pick"
```

and another window appears - giving you a listing of all jobs with open CONsoles which can be picked to the top of the pile of windows.

Next, try

```
EXEP "Button Frame"
```

... nothings happens. You may have noticed an access to your floppy and/or harddisk. "Button Frame" is not an executable Thing, therefore the EXEP command gives up on Things and tries to open a file called "Button Frame" on the currently defined program default device (can be set with PROG_USE).

Okay, let's go back into the "Things" menu and click on the word "Things" in the list. The window will change and you can see that there is (at least) one job registered as the user of this Thing, which is, hardly surprising, "Things". Confusing? Press ESC to get back to the list and select Pick. This should be less confusing.

If you have buttons in the Button Frame of QPAC2 then you can look which job registered itself as a user - just go back to the Things list and select "Button Frame".

For the curious ones: The "Button" Thing is a Thing with Extensions you can see them in the detailed Thing window if you select "Button".

QPAC2 comes with many Things but there is one problem: most Thing names are the same in different languages, but some are not. We, the translators, (Wolfgang Lenerz for the French version and me for the German) decided indepently of each other that "Button_Sleep" does not mean much to a German or French user and translated it. Big problem! From then on, it is not possible to replace an English QPAC2 by a German one and vice versa. But I guess it is too late now to change things (or Things?) and, for example, "Dateien" back into "Files". The solution is probably to have both the English and German (or French) Thing name for the same job ... but this is a different matter.

Let's create a new Thing. You want to turn XCHANGE into a Thing like the "Pick" menu? Easy, the HOTKEY System helps you doing it:

```
ERT HOT_CHP ("X","XCHANGE")
```

You should have XCHANGE available on your current program default device. It is copied into memory and turned into an executable Thing, named XCHANGE. Check the "Things" menu, it is there!

And, even better, if you now type

```
EXEP "XCHANGE"
```

the program will be started without any disk or harddisk access! Of course, it can be started as well using the HOTKEY System II (by pressing ALT X but remember to get the HOTKEY Job going by typing `HOT_GO`first!).

When you remove the HOTKEY definition, the Thing will be removed as well.

The HOTKEY System II provides other functions to allow HOTKEYs to be assigned to existing Things which are not removed when the HOTKEY is removed, but this will be explained in one of the next part of this series. Also, the Things built into QPAC2 are more advanced than the Things you can create manually, but this is material for another article too.

■

Q-Help

D'you Need Somebody?

Roy Wood

Before I start this review I would like to detail the relationship between myself and Rich Mellor, the programs author, and the degree to which I have been involved in the program and the reference manual. Q Branch have been involved with Rich for some time and we publish and sell his Q-Route program. Rich came to us with the Reference Manual in 1996 and we spent some time together turning the plain text files into the finished article. By the time the manual was ready for release Rich had re-formed his old company, RWAP Software, and having added up the costs of producing the manual we decided it was better if Rich went ahead and sold it himself. Since then he has added quite a few programs to his catalogue and Q-Help is sold purely by RWAP.

We do offer some of RWAP's software and the Reference manual in our catalogues but this arrangement is purely because they have no credit card facilities whereas we do. I make no profit on these sales and I redirect all cheques directly to RWAP.

History Lesson

OK, now that's out of the way, it's time for a little history. When Rich and I first delivered the Reference manual to a hungry public at the Hove show in 1998 there were immediate cries for it to be available in an electronic format. During its conception we had considered this but but rejected it on the grounds that an electronic format did not suit the use to which we expected the buyers to put it.

Ok, it was never going to bedtime reading but it was more than just a list of KEYWORDS and SYNTAX. We expected, and most of the purchasers have bourne this out, that people would use this as a means of planning out programs away from the keyboard and to browse through the lists of new commands available to users in the extra toolkits.

We did realise, however, that putting all the keywords on a searchable disk would be a good idea and that became the next project. This is where I bow out of the story except for the occasional bit of Beta testing and a few suggestions.

System Requirements

The program was tested on four systems: Q 40, Aurora/SuperGoldCard/Qubide/superHermes, 200MMX PC laptop running QPC 1, 350 MHz AMD desktop PC running QPC 2.

You will need a minimum of a Gold card QL and HD drives because the files do take up a lot of space and there is no option to span the data files over several disks. It would install onto a 4 Mb ROMDisq (and be very fast from that medium) or a Qubide/Hard Disk or Emulator/Hard Disk system. A Gold Card system would only be needed for access to drives of higher density than a DD, memory is not an issue. Systems with access to screens larger than the standard 512 x 256 would be able to see more of the text area at one go and take advantage of the split screen feature.

Overview

Q-Help is designed to be a hypertext style reader with access to all of the keywords in the manual. It does not have the depth of the manual in that it does not go so far into the various differences between processors and it does not have the lengthy sections on hardware expansions and programming hints. It does not need a hypertext reader program because it has its own interface to the text.

The program is supplied on a single DD disk but does need to be installed onto a larger medium before it can be used. The help files themselves are supplied in zipped format as are the actual program files. The BOOT file is an installation program which unzips the files to the target directory using the INFOzip unzip v 5.40 (also supplied). There is no written manual for it, as such, but the installation program runs smoothly and the program is very intuitive to use.

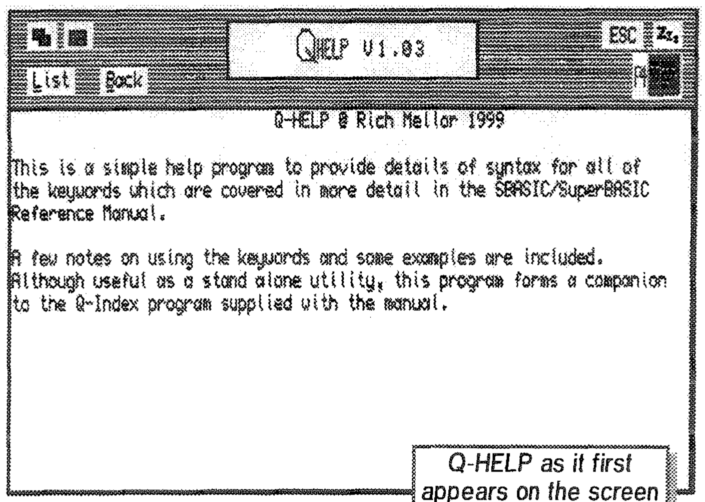
What You Get On The Disk

Boot - The BOOT File (Obvious really)
Manual_doc - The Manual in Quill Doc Format
prices_doc - RWAP Price List (Bit of an advert)
QHelp_files_zip - The help files themselves
QHelp_zip - The Program files
Config_exe - The standard configuration program
hot_rext - The P.E. Files
PTR_GEN - "
WMAN - "
qhelp_obj - The Main Program
QLIB_run - QLIB runtimes
UPDATES_txt
sigext30_rext - The Signal Extensions (see below)
unzip - The unzip program

Onto the Installation

Put the disk into flp1_ and type 'LRUN FLP1_BOOT'. The first question it asks you is where the ZIPped file is so enter 'WIN1_' the next quest ion it asks is where you want it to be installed to. Assuming that you are installing onto a hard disk enter 'WIN1_'. You can re-direct this to a subdirectory of your choice but, if you choose the root then the program will create a subdirectory for you called 'WIN1_BASIC_' and install itself into that creating other subdirectories as it needs them.

The first version of this program that I received from Rich did not load the supplied signal extensions in its boot but that should have been fixed in later versions. I am not too sure why the signal extensions are needed because the program installs with no problem when they are not loaded but when the installation is finished you will find an error message stating ******* Signal Extensions Not Loaded ******* I deleted the installation, installed the Signal Extensions and reran the boot. There appeared to be no difference in the programs which were installed or the way the programs ran and the only difference I saw was a distinct lack of error message. I seem to recall that this is something generated by the unzip program and so should not be laid a Rich's door.



Q-HELP © Rich Mellor 1999

This is a simple help program to provide details of syntax for all of the keywords which are covered in more detail in the SBASIC/SuperBASIC Reference Manual.

A few notes on using the keywords and some examples are included. Although useful as a stand alone utility, this program forms a companion to the Q-Index program supplied with the manual.

Q-HELP as it first appears on the screen

One only other extension required for the program to run is neither loaded nor supplied on the main part of the disk and that is QLIB_RUN. Most people will already have this installed on their systems and, although it is not needed for the installation process it is required for the program itself and can be found in QHELP_zip.

And Now the Program Itself

Before you start you should check the program configuration using either the CONFIG_EXE program supplied or JOCHEN MERZ's MENUCONFIG. Since there is only one config item in the block for v 1.03 this is a swift task. That item refers to the place where the help topics are found and, if you installed to WIN1_ this is set up for you. No problem there then.

There are three ways you can use this package but I will describe the easiest option first along with the workings of the program, itself. After that I will tell you the alternatives.

I will assume, for the sake of simplicity that the program is installed into 'WIN1_', if you have put it somewhere else change the device accordingly.

1. EXEC IT!

First we have the simple method. After you have loaded the QLiberator Extensions you can just EXEC the program in the normal way. You can do this from the command line with a simple:

```
EXEC win1_basic_qhelp_obj
```

You should see a screen with the heading 'QHELP v 1.03' This screen is the standard 512 x 256 size so it will run on any QL system with enough memory It is divided horizontally into two rectangles. The top part is in the red and black striped option available as one of the four standard Pointer Environment colour schemes (although there is no option to swap it for any of the other schemes). This part contains the controls for the program. Starting at the top left hand corner there are the two standard P.E. 'Move' and 'Resize' icons. in the opposite corner there is a box marked 'ESC' which will allow you to quit the program and the standard 'Sleep' button which will reduce it to a button if you have the QPAC 2 button frame loaded. If you 'HIT' (left mouse button) the 'RESIZE' icon you manually change the size of the overall window. If you 'DO' (right mouse button) the same icon the program will resize itself to fill the screen.

Below this there are three further icons:

LIST, BACK, and a picture of a split screen with F4 beside it

LIST: This gives you a pull down menu which accesses the available help files. There are three icons available in this menu.

'ESC' allows you to quit back to the main part of the Program 'A-Z' allows you to sort the keyword list into alphabetical order. 'MOVE' allows you to move the box but only within the window of the main program.

When the list is first loaded it is not a complete alphabetical list but is sorted into three sections 'DIY Toolkit', 'BASIC HELP' and 'TK2' If you want to view the list as a whole, click on the 'A-Z' icon and this will sort the list into alphabetical order.

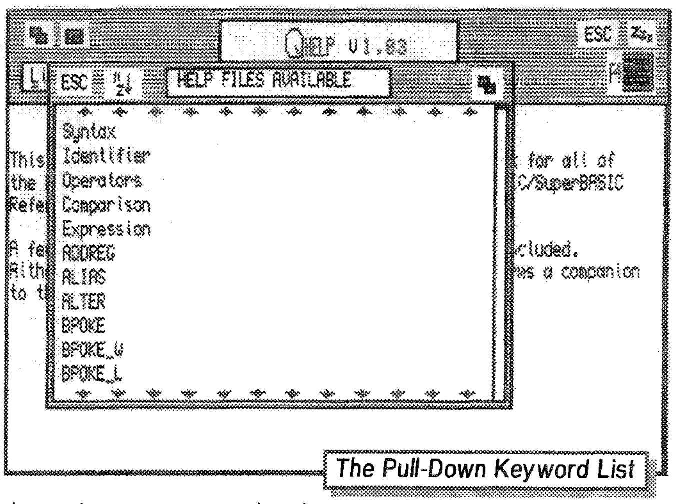
To view a file from the list simply click on the file name and the file is loaded onto the screen. The files are arranged in the following way:

At the top you will find the KEYWORD or KEYWORDS that are covered by the description. Bellow this you will find a description of the SYNTAX for the KEYWORDS and below that the 'Location' (ie. where the KEYWORD can be found. That is to say which toolkit it is in or which ROMs have access to these commands). These are followed by a description of the actions performed by the commands and hints on how to use them, including, where possible, examples.

At the bottom of the 'page' you will find a section titled 'CROSS-REFERENCE' and this is the clever bit. As you move the cursor over the 'CROSS-REFERENCE' section you will find that it changes from a standard pointer to a 'pointing hand' and, if you 'HIT' (left mouse button) the word that it is on when the change occurs you are immediately taken to that file.

Another useful function is the ability to stuff any line of text into the stuffer buffer. Just move the pointer onto a line and 'DO' it (right mouse button). You will then get a new menu window which will ask if you want to 'PLACE THIS LINE IN STUFFER BUFFER'. There are two buttons above this marked 'YES' and 'NO' and these will, unsurprisingly, either accept or reject the action.

All of the above actions can be taken without the use of a mouse since a 'HIT' can be done by the SPACE BAR and a 'DO' by the ENTER key. All menu items can be accessed by pressing the letter which is underlined.



The Pull-Down Keyword List

BACK: This icon allows you to retrace your steps in case you have been led into a 'dead end'. If you follow a trail of links and find that you have made a wrong choice each time you click on the 'BACK' icon you will be taken one step back along the trail.

F4: This is an very interesting icon and allows you to change the view of the main window. When the program is first loaded the area beneath the Command section is a single box with scroll bars top and bottom. Access to the list is only via a pull down menu. If you have a screen resolution greater than the QL standard 512 x 256 pixel display you can activate the second method of viewing the program.

Click on the 'F4' item and you get a split screen view in which the list is permanently displayed on the left and the text on the right. This makes it very easy to select a keyword from the list and view the text. The reason that you cannot use this from a standard display is merely that there is not enough room on the screen to do it. There is one thing which I think could be added to this view and that the possibility to link the KEYWORD list on the left to the text screen on the right so that following a link to a cross reference would also scroll the list to centre on that word as well. This would allow you see other KEYWORDS with similar names.

Having described the program's functions I would now like to mention one other way that the program can be used and another way to access the program's data.

LRESPR IT!

You can load the main program as a resident procedure and call it up directly from the command line. For instance load the main program by typing:

```
LRESPR WIN1_BASIC_QHELP_OBJ
```

Then type:

```
Q_HELP
```

at the command line and the program will appear and load the opening file.

Alternatively type:

```
Q_HELP 'EXEC'
```

Once more the Q-HELP window will load but this time it will display the file for the keyword 'EXEC'.

There are some drawbacks to this method of working however. The first is that the BASIC command line does not like to return control to the user when you quit the program. You will find yourself looking at the command line you just typed but with no flashing cursor. The answer to this is to press ‹CONTROL-SPACE› at the same same time and this will allow you to regain control of BASIC.

The second drawback is a bit more fundamental in that typing part of a command will not allow the program to find the appropriate text. For instance, typing:

```
Q_HELP 'SELECT'
```

will result in an error message as will:

```
Q_HELP 'SELECT ON'
```

whereas:

```
Q_HELP 'SELect ON'
```

will find the correct file. You really have to know what it is that you want to find. Apart from that this is a very good way to use Q-HELP and can be a great help when working on a complex program. One tip here for SMSQ/E owners is to open an SBASIC window and work from there because you can then 'CONTROL-C' between this and the BASIC file you are working on.

CALL QD!

Ah, my favourite text editor. QD already comes with a hefty Hyper-Help file of its own but the files from this package are far more extensive. All you need to do to get QD to use Rich's help files is to point it in the right direction. There are three ways to do this. If you want it to be permanently set to use the Q-HELP file go to the config block in QD and then to the section labelled 'QD THINGS/FILES' there is an item there which will set the help file to be called so just point this at 'WIN1_BASIC_HELP_' and it will use them.

QD also offers the option via the 'STATUS' menu in the main program. Press 'F5' within QD and the status menu pops up and from there you can change the help file settings. For my own purposes I always load QD as a resident extension and then create distinct 'THINGS' which will serve a special purpose.

In my BOOT file I have these lines:

```
940 ERT HOT_THING ('S','QD';'\U B\e _bas
    \t SBAS/QD\D WIN1_MYPROGS_
    \h WIN1_BASIC_HELP_\|0')
950 ERT HOT_THING ('B','QD';' \U B\e _bas
    \t QBASIC\D WIN1_MYPROGS_
    \h WIN1_BASIC_HELP_ \|0')
960 ERT HOT_THING ('H','QD';' \U B\e _bas
    \t QBASIC\D WIN1_MYPROGS_
    \h WIN1_QH_BASIC_HELP_ \|0')
```

I have installed Q-HELP to WIN1_QH_ so the following lines perform these actions:

940 : Call QD with the User option set to 'BASIC', the extension of the file set to '_bas', the F10 option set the SBAS/QD THING (part of SMSQ/E), the help file directory set to WIN1_BASIC_ (Hyper Help Files) and the word wrap option turned off.

950 : Call QD with the User option set to 'BASIC' and the extension of the file set to '_bas' but the F10 option is set to 'QBASIC' which is the link between QD and QLIBERATOR. all other options are the same.

960 : This has all of the same options as LINE 950 but this one will call the Q-HELP help files instead of the standard D ones.

The same drawbacks to the use of the files applies here however in that asking for help on 'SELECT' will not find anything in the Q-HELP files whereas Jochen's Hyper Help will correctly lead you to 'SELECT ON'.

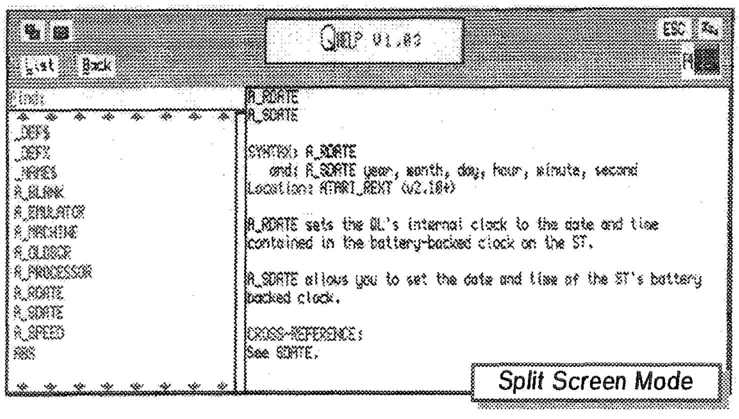
Writing BASIC programs in QD is different to writing them in the Superbasic editor because it does not correct KEYWORDS like REMARK to REMark or phrases like 'SELECT ON' to 'SELect ON'. This can confuse Q-HELP altjough HYPER HELP-has no problems with it.

QINDEX

Of course Rich couldn't resist linking Q-Help to Q-Index and adding yet another useful level to it. Q-Index is the index program that you will find on disk one of the three disks you get with the SuperBasic/SBASIC reference manual. It acts as a contextual, electronic index for the manual so you can look up a subject and then get a list of keywords related to that subject.

The clever part here is that, if you have LRESP'd Q-Help and have version 1.02 of Q-Index you can link the two programs together by changing a 'config item' in Q-Index. Next time you fire up Q-Index you will find that most of the KEYWORDS are in White and when you click on them you will get a Q-Help window opening up with the definition in it. You can still navigate around Q-Help in the way described in the main section of this review and when you hit 'ESC' you are returned to the Q-Index window and can carry on from where you left off.

It is a shame, in a way, that you cannot get back to the Q-Index screen while Q-Help is displayed because I can see the point of being able to open two or three Q-Help windows with different definitions in them but this is a minor niggle.



```
A_RDATE
A_SDATE

SYNTAX: A_RDATE
   and: A_SDATE year, month, day, hour, minute, second
Location: ATARI_REXT (v2.18+)

A_RDATE sets the QL's internal clock to the date and time
contained in the battery-backed clock on the ST.

A_SDATE allows you to set the date and time of the ST's battery
backed clock.

CROSS-REFERENCE:
See SDATE.
```

CONCLUSIONS

As a programmers reference tool Q-HELP cannot be beaten. It is versatile and a very clever piece of programming. At £10.00 the price is ridiculously cheap and it will probably be very useful when when decoding those odd bits of BASIC that you come across in the QUANTA library or other PD files. For the BASIC programmer who wants to extend his range it is also a useful tool and it really becomes helpful when you have a screen wider than the standard QL one so you can have it and an editor running side by side. It does not replace the SBASIC/SuperBasic Reference Manual because that is an in-depth work but it does give you a handy functional aide to writing neat programs using all of the tools available to you. Well worth a look.

QL-PC Transfers

Geoff Wicks

Before he sent his review of Style-Check 3 to QL-Today, Henry Orlowski kindly telephoned me to tell me of his main conclusions. When he read out one of his suggested improvements to the program, he was surprised by a hollow laugh from my end of the telephone.

The suggested improvement was an "ability to resave in the original (word processor) format". When Henry phoned I was in the middle of writing a QL PC word processing file transfer utility, and had spent many hours unsuccessfully trying to discover the file formatting in QL Word processors.

The 3 main QL word processors format their text in a similar way. The file starts with a block of identification, file length and header and footer information. Then comes the text in which control codes for such things as bold and underlined text are included. (Text87 stores its text in a slightly different way.) Finally there is another block of formatting information that remains a mystery. Presumably it contains tables with details of justification, margins and tabs.

As far as I know, little or nothing has been published over the detailed formatting of QL word processing files. Does any one have any information? More challenging, has anyone ever written a pseudo-Quill file? If it were possible to do this, it could also be loaded into the other QL word processors.

By the time this appears, Just Words! will probably have published a QL PC transfer utility. From the QL to the PC this program will transfer only global formatting information and not detailed changes. From PC to QL texts can be transferred only as ASCII code. It would be interesting to have a program to convert Rich Text Format into Quill format.

While I have been researching and writing this transfer program, QL-ers have suggested other facilities they would like. Examples are a filter to extract the text from e-mail files, a program to remove unwanted spaces from OCR read text and possibly a program to extract the text from Microsoft Word files. At present QL-ers are doing this slowly and manually using text editors.

Do you have similar or other needs? If so I would like to hear from you as I am making an inventory of the types of transfer being made from QLs to PCs and vice versa.

E-mail: geoffwicks@hotmail.com

■

SUQCESS

A database-manager

by Wolfgang Uhlig

An Announcement

During recent years I've had a lot to do with databases. I've used mainly the DBAS-suite by D. Howell, debugged and improved by Phil Borman. Together with the DBAS database engine comes a small program to access and manipulate DBAS-databases. It is not very comfortable and somehow old fashioned, just enough to do the most necessary of things. Thus the wish for MORE came up, I wanted to have a program where a few mouse- or/and key-clicks are enough to manage the important functions of a database. I wanted a better overview of the data, a modern, nice looking frontend for various screen resolutions, a connection with for example QD or QSPREAD via the srap and of course modern capabilities like the menu-extension. In other words, a state of the art database-manager for the QL.

"But such a program doesn't exist!"

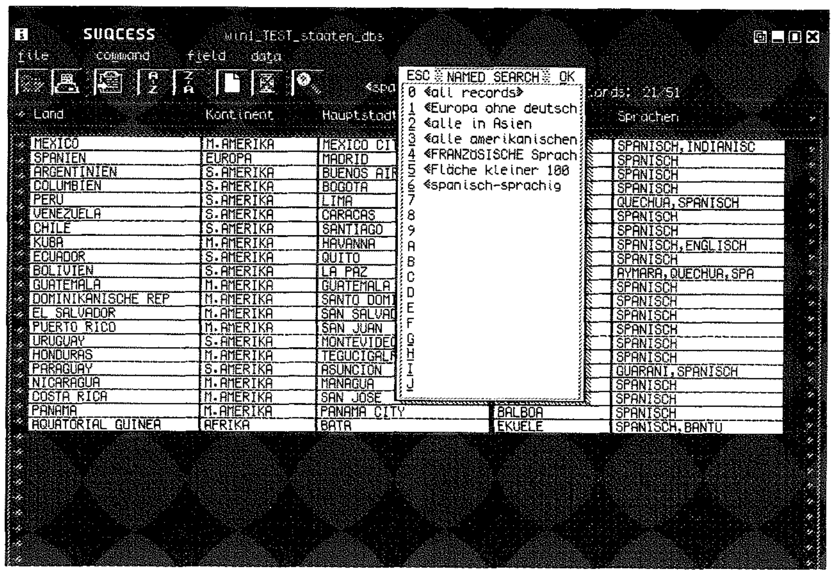
Or rather, it didn't exist because I decided to write it myself. It has been a long task to do and it is not completely ready, either, but nevertheless: "It does exist now!"

Its name is an allusion to the professional program ACCESS and to the fact that for the QL, too, beautiful programs can be written SUCCESSfully. (Of course I know that trying to compare it with Access would be absurd!)

Suqcess is as far as I know the first fully pointer-driven database program for modern 'QLs'. It includes Jochen Merz's menu-extensions, shows different menus made with EASYPTR (by Albin Hessler), uses drag and drop routines, has several screen resolutions and much more. Its functionality reminds of QD and QSPREAD, that means, every function can be invoked via menus, submenus, icons or simply by keyboard keys.

Contrary to all other QL database programs I know, SUQCESS doesn't show records separately, but as a kind of spreadsheet. I chose this form because I always had been annoyed of being unable to compare records directly. Having made a search, I hate it to see that there are 20 records but I will have to look at each of them until I've found the right one. With SUQCESS this is over, at a glance you see all found records or at least the first fields of them. What is more, you can choose one or more records and copy it/them to the

scrap or to a file. You also can print them, setting which fields you want to be printed and in which order. All this is done by drag and drop and several mouseclicks!
When sorting or searching, it is no longer necessary to write any command. You 'drag' the right operator and 'drop' it near the search field (invalid operators are refused by the program). Then you fill in the searchstring or number. (See picture 5). A click on OK will start the search. At the moment you can connect up to three search conditions (with AND, OR and XOR will be achieved later). To avoid re-typing, SUQCESS gives you the possibility to save the searches you've done and to give them a name. Thus it is possible to redo a search in the quickest possible way! Of course you can rename or delete them if you want to.



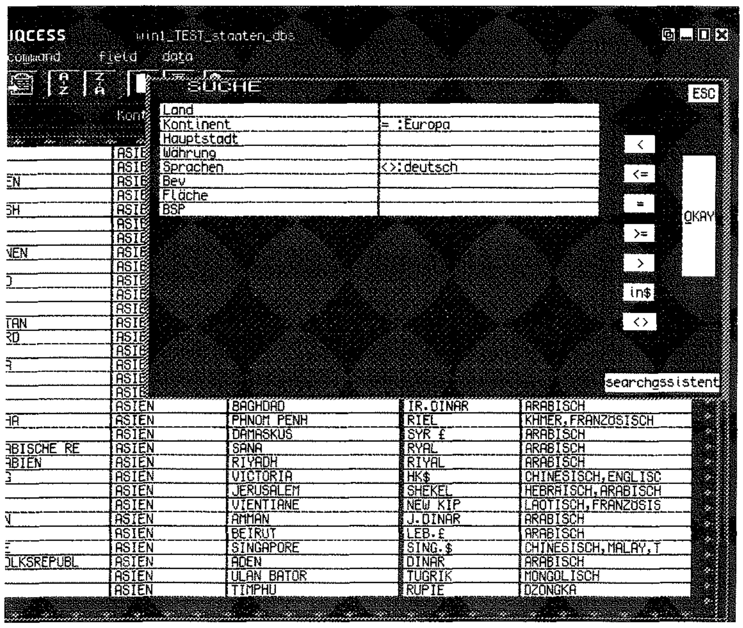
```
SUQCESS          win1_TEST_staaten_dbs
file   command   field   data
                                   ESC  NAMED SEARCH  OK
                                   0 «all records»
                                   1 «Europa ohne deutsch
                                   2 «alle in Asien
                                   3 «alle amerikanischen
                                   4 «FRANZÖSISCHE Sprach
                                   5 «Fläche kleiner 100
                                   6 «spanisch-sprachig
                                   7
                                   8
                                   9
                                   A
                                   B
                                   C
                                   D
                                   E
                                   F
                                   G
                                   H
                                   I
                                   J
«spa                                          ords: 21/51
Land                    Kontinent    Hauptstadt            Sprachen
MEXICO                  M.AMERIKA    MEXICO CI             SPANISCH,INDIANISC
SPANIEN                 EUROPA       MADRID                SPANISCH
ARGENTINIEN             S.AMERIKA    BUENOS AIR            SPANISCH
COLUMBIEN               S.AMERIKA    BOGOTA                SPANISCH
PERU                    S.AMERIKA    LIMA                  QUECHUA,SPANISCH
VENEZUELA               S.AMERIKA    CARACAS               SPANISCH
CHILE                   S.AMERIKA    SANTIAGO              SPANISCH
KUBA                    M.AMERIKA    HAVANNA               SPANISCH,ENGLISCH
ECUADOR                 S.AMERIKA    QUITO                 SPANISCH
BOLIVIEN                S.AMERIKA    LA PAZ                AYMARA,QUECHUA,SPA
GUATEMALA               M.AMERIKA    GUATEMALA             SPANISCH
DOMINIKANISCHE REP      M.AMERIKA    SANTO DOMI            SPANISCH
EL SALVADOR             M.AMERIKA    SAN SALVAD            SPANISCH
PUERTO RICO             M.AMERIKA    SAN JUAN              SPANISCH
URUGUAY                 S.AMERIKA    MONTEVIDEO            SPANISCH
HONDURAS                M.AMERIKA    TEGUCIGALP            SPANISCH
PARAGUAY                S.AMERIKA    ASUNCION              GUARANI,SPANISCH
NICARAGUA               M.AMERIKA    MANAGUA               SPANISCH
COSTA RICA              M.AMERIKA    SAN JOSE              SPANISCH
PANAMA                  M.AMERIKA    PANAMA CITY  BALBOA   SPANISCH
AQUATORIAL GUINEA       AFRIKA       BATA         EKUELE   SPANISCH,BANTU
```

Making new or deleting records works on mouseclick. Deleted records are saved in a (for each SUQCESS-database created) undo-file and can be restored - a service for the thoughtless users among us :-))
Creating a database is done interactively, don't worry about it, just do it! You can import PSION-export files and ASCII-files in a simple format that is explained in the manual and of course you can export files in a format that can be imported by all 'big brothers' out there. SUQCESS can be configured to meet your own wishes. Up to now you can determine:

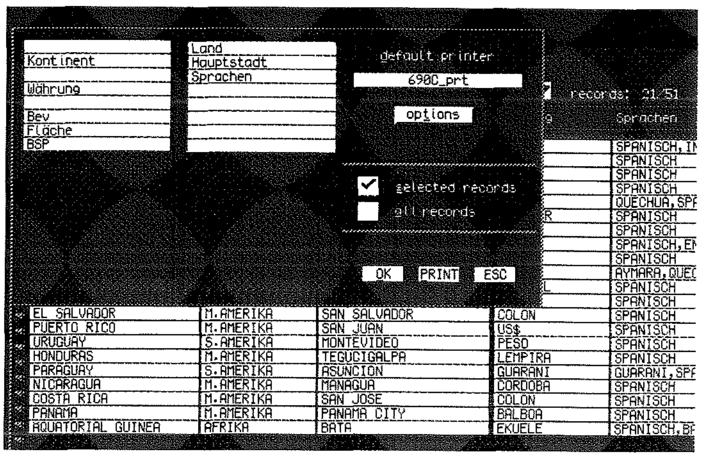
- which screen resolution you want to have at the start of the program (512x256, 640x480 or 800x600), whereby the program automatically recognizes whether this is possible whatsoever.
- How many records you want to be visible (the more you want the slower it works).
- How wide you want to see the columns (in small columns the contents is virtually cut off but you can see it if you click on it).
- The directory for your databases
- Whether you want LONG INT-fields to be interpreted as DATE-fields or not (This is useful if you like to save exact dates but want to see the date as a string, for example).
- Which printerdriver should be used.

The last is still a sort of tricky thing. At the moment I use two small programs I once wrote, one for the HP Deskjet and the other for an OKI which is Epson compatible, by calling them in the print preset routine. I hope that there will be a better solution in the future of the QL.
SUQCESS is not programmable Archive-style, but that wasn't my intention, anyway. It is a program which allows you to manage data in a simple and intuitive way. Don't worry about wrong syntax or expressions, just click and off it goes!
There are, ultimately, some restrictions:

- SUQCESS is a pure SBASIC-program, using the machine routines of DBAS, MENUREXT and EASYPTR. It is therefore not as fast as a machine program. Even then it seems quite good with databases that are not too big (not more than 5000 records) and when run on a QXL for example.
- There is an unexplored mistake in the MAWDRAW-routine of EASYPTR when trying to show a three dimension field with more than 1265 'records'. I therefore restricted the visible records to 1250 which should be not a big problem. You



```
UQCESS        win1_TEST_staaten_dbs
command   field   data
          SUCHE                                        ESC
          Land
          Kontinent       = :Europa
          Hauptstadt                                    <
          Währung                                       <=
          Sprachen        <>:deutsch                    =
          Bev                                           >=     OKAY
          Fläche                                        >
          BSP                                           in$
                                                        <>
                                              searchassistent
              ASIEN    BAGHDAD      IR.DINAR   ARABISCH
HA            ASIEN    PHNOM PENH   RIEL       KHMER,FRANZÖSISCH
              ASIEN    DAMASKUS     SYR £      ARABISCH
ABISCHE RE    ASIEN    SANA         RYAL       ARABISCH
ABIEN         ASIEN    RIYADH       RIYAL      ARABISCH
G             ASIEN    VICTORIA     HK$        CHINESISCH,ENGLISC
              ASIEN    JERUSALEM    SHEKEL     HEBRAISCH,ARABISCH
              ASIEN    VIENTIANE    NEW KIP    LAOTISCH,FRANZÖSIS
N             ASIEN    AMMAN        J.DINAR    ARABISCH
              ASIEN    BEIRUT       LEB.£      ARABISCH
E             ASIEN    SINGAPORE    SING.$     CHINESISCH,MALAY,T
OLKSREPUBL    ASIEN    ADEN         DINAR      ARABISCH
              ASIEN    ULAN BATOR   TUGRIK     MONGOLISCH
              ASIEN    TIMPHU       RUPIE      DZONGKA
```



Kontinent	Land			
Währung	Hauptstadt			
Bev	Sprachen			
Fläche				
BSP				

				SPANISCH,I
				SPANISCH
				SPANISCH
				SPANISCH
				QUECHUA,SP
			R	SPANISCH
				SPANISCH
				SPANISCH,E
				SPANISCH
				AYMARA,QUE
			L	SPANISCH
				SPANISCH
EL SALVADOR	M.AMERIKA	SAN SALVADOR	COLON	SPANISCH
PUERTO RICO	M.AMERIKA	SAN JUAN	US$	SPANISCH
URUGUAY	S.AMERIKA	MONTEVIDEO	PESO	SPANISCH
HONDURAS	M.AMERIKA	TEGUCIGALPA	LEMPIRA	SPANISCH
PARAGUAY	S.AMERIKA	ASUNCION	GUARANI	GUARANI,SP
NICARAGUA	M.AMERIKA	MANAGUA	CORDOBA	SPANISCH
COSTA RICA	M.AMERIKA	SAN JOSE	COLON	SPANISCH
PANAMA	M.AMERIKA	PANAMA CITY	BALBOA	SPANISCH
AQUATORIAL GUINEA	AFRIKA	BATA	EKUELE	SPANISCH,B

won't want to have an overview of more than 1000 records, will you? It concerns only the visibility, of course, database actions such as search, sort, delete etc. happen in the database itself and are not affected at all.

• The front-end-design of SUQCESS leaves less room for data, so a resolution of 512x256 actually is too small for a good overview. Higher resolutions are much better.

• Since most actions read/write to/from a drive, either a lot of time or a hard disc will be necessary. Also, especially for import routines, you should have enough RAM, at least 2 MB, I guess.

• The program has been developped under SMSQE and could give problems under other OSs. I tried it with Minerva 1.97, which worked quite well, exept for some small problems with a PAN-action. But, even if some people are likely to hate me, I don't intend to follow the philosophy that any program must run on any QL even it's a dinosaur. Sorry for that!

All in all I believe that SUQCESS will be a program which can be useful for a lot of QL-users. It is not completely ready yet, but I intend to have a 99% version in the beginning of october and to be able to sell it later this year. Look at the ads of Jochen Merz.

If anybody is interested in having a trial version, please e-mail me and I'll send a zip-file containing everything you need for a test. All functions will be available then, creating or importing a database, however, will be only virtual.

Wolfgang Uhlig
De Karn 28
NL - 6581 WJ Malden
Netherlands
e-mail: wolwol@compuserve.com

■

BYTS OF WOOD

SAW POINTS OFFCUTS AND SNIPPETS

First off this month an apology to Phil Borman. His BBS has not been taken off the air he has just moved to a new phone service and the number has changed. You can find the Nene Valley BBS at

01933-460538

His website can be found at:

http://www.pborman.freeserve.co.uk

My apologies to Phil for disseminating the wrong information.

Its a standard - its just the flagpoles are slightly different

For many people the reason that they drift off into using a PC is the 'information interchange factor' They want to be able to swap information with other computer users and find that, since many of the other people do not use a QL, the QL formats are incompatible with the PC ones. I mentioned in the last issue that Geoff Wicks has been busy writing a new utility which will take files from the QL wordprocessors and convert them into 'RTF' (Rich Text Format) In the course of looking at this piece of software the lack of viable standards on the PC comes into sharp focus.

Take a Text 87 file and run it through Geoff's software. Take the resulting file and insert it into 'Write' (Written by Microsoft - part of Windoze). The result is a text file which is perfectly readable and with all of the format intact. Take the same RTF file and import it into 'Word' (Written by Microsoft and part of Orifice 97). The result is an error message stating the file is not in Rich Text Format. Compatible eh?

My wife is part way through a large translation job. She is translating the latest Michelin Guide to New York into German using Word from the Orifice 97 suite. The recipients of the finished work actually want it in Word 6 format. Word provides you with a little add on feature which allows you to save and import files from other formats including earlier versions of Word itself. However, when she does this the resulting file is subtly changed. Superscripts become subscripts symbols alter and the bold headers go. Since this software is all written by same software house and published by the same company you

would think that they might get it right - well maybe not.
Geoff told me, at the Eindhoven meeting, that he hopes to have his QL2PC program released in time for the Paris meeting which will have happened by the time you read this I have a new version of the beta test file which looks very interesting but I have had little time, so far, to look at it.

DATAbase Diary

In last months Quanta magazine there was a short article on how one user translates Archive export files for use on a PC. Just by chance Stephen Hall made a discovery just before this was published which may be of interest in this area.

First a little history

As the UK distributor for QL Today I hold a list of users in a DATAdesign database. I like the flexibility of DATAdesign and I really like the way it can be accessed from BASIC. Steve, on the other hand, has always preferred to use Flashback. When it comes to issue time I prepare an export file from DATAdesign and pass it to Steve who then imports it into Flashback. He then runs a flashback script which prints the labels for the envelopes. This originally came about because he had a dot matrix printer which would accept the tractor feed label sheets.
When I gave him last month's file he was in the midst of looking at Micro$oft's Access database because he was producing a system for a club his son belongs to. He wondered if he could import the file I had just given him but he got one giant record when he tried to do it. Being of an analytical mind he wanted to know why so he loaded the file into a program called the Programmers File Editor (PFE). This is a free PC program which boasts a whole range of features and utilities. At the bottom of the text window there is a small toolbar which has a small button related to the file format. If you look at this when an '.exp' file is loaded you will see it says 'UNIX' indicating that the file is in Unix format.
Now the clever part. Click on the box and this changes to 'DOS'. Save the file with a new file name and it is now in 'DOS' format. The difference is slight but very important. The record separators have both a line feed and a form feed at the end of them whereas the UNIX ones do not have the line feed. This means that the resulting file can now be separated into records.
One final thing baffled me when I tried to use the process he explained to me on the telephone on my system. Try as I might I could not get the information into the Access database. At first the problem was that I had not loaded the import filters for Access. In true Windoze style these are not loaded when the program is installed nor are they an option available from any of the menus on the main program. They are, in fact, only available from the help file. Go to the help file, look up 'import' and then click on the icon in the text to install them from the original CD. Even then I could not get the system to work. The final key was the file extension. Access, it seems, rejects the file because it does not recognise the '.exp' extension. Change this to '.txt' and it works fine. Now that is really dumb!
Then came a new revelation. I told Jochen about this on the telephone and it obviously did not really sink in until I sent him the article to go into the magazine. I got an immediate telephone call from him 'QD already does this', he said. I tried it and found that it does it automatically. If you load an export file into QD and then save it onto a DOS disk as a text file it adds the carriage returns for you and the resulting text will load into a PC database just as easily. Apparently this is mentioned in the manual and has existed as part of QD for a long time but I have owned and used QD for over seven years and this one slipped by me completely. Great, another use for my favourite file editor!
The Programmers File Editor is a useful addition to any PC system and exists in two formats - one for 32 bit systems and the other for older systems. You can get it on the internet by going to:
http://www.lancs.ac.uk/people/
click on the authors name - Alan Phillips - and follow the links form there. It is freeware.

Another DATAbase

Also seen at Eindhoven last month was a program built around the DBAS DATAbase engine. Daniel Baum has been working on a pointer driven front end for this program for a while although a lot of his time has been taken up with a new child. I have been the recipient of his efforts so far and the result is a very worthy piece of programming although not quite ready for commercial release.
The front end that I saw at the Eindhoven show is also a very good piece of work with some very interesting features. I don't know if there are any commercial plans for the release of this program but I really feel that there should be. Both programs were produced in BASIC with the pointer driven parts added in by use of EASYptr. It may seem, since I sell this pro-

gram, that I am engaging in a little free publicity here but with a little practice and a bit of help from Norman Dunbar and John Miller's Pointer Environment Kit (PEK) tutorials it can produce some excellent results. Anyone wishing to get a copy of PEK can try the BBS, the PD suppliers, or as a last resort myself. Q Branch has been supplying the PEK as a free disk with EASYptr and most of the people who have worked their way through the examples have found it to be an eye opener. Good work Norman and John.

And if they can do it..

Finding good programs like this, often from people who think that they are just ordinary users, makes me think. There is a wealth of potentially good and very useful software which people are very shy about putting before a wider audience. Pete Marsh came into the shop the other day to upgrade his copy of QPC to the new QPC 2 and he showed me something he had written.
Pete works on a dairy farm and had written a whole suite of programs to assess the milk flow from the herd and perform all the feed calculations needed to run the farm. Things like this are possible because the QL system is so flexible. They take a little time to develop and they need a degree of patience but they are achievable by anyone who takes the time to try it out.

Sounds OK

Another piece of software which Q 40 owners might like to get their hands on is AWAVE. This is available from
http://www.fmjsoft.com
and is shareware. Of course this is yet another Windoze program but you can use it to convert almost any sound file into one that will play on a Q 40. The software itself is quite simple to use and will do a whole lot more than converting sound files for the Q 40 so check it out.
If you think this is all becoming a bit too PC oriented then I can only say that if you are going to use the PC you should at least support the small software houses and authors and not sling all of your cash at Micro$oft.

Did you Put the Trash Out?

One of the new features that appeared a while ago with the advent of version 2 of the Qubide ROM was the 'Trashcan'. This allows you to place files you have deleted from your hard disk into a special section with is tagged as a waste basket. The files are not deleted as such until you 'empty the trash can'.
This may, on the face of it, seem a pointless thing because if you delete a file what you want to do is to throw it away and not keep it but I know that I have sometimes been on a 'clean up binge' only to find that I have accidentally deleted a file that I should kept. Of course there are the backup disks and I do have two 'Syquest' drives connected to my systems on which I back up whole partitions but the trashcan is a quick and useful way of retrieving a single file that you have accidentally thrown away. After all, how many times have you had to go to the waste bin in your office to retrieve a crumpled piece of paper with an important telephone number on it ?
Unfortunately there is a bit of a glitch in the cybernetic rubbish bin. Rich Mellor reported this to me this month and it is something that you should look out for. The problem seems to relate to the way that some programs update files. I found once that there were a whole lot of files in my trashcan which I did not remember ever deleting. Emptying the trash can cause a loss of files on the system. Luckily backups came to the rescue but I never managed to work out what had happened. I asked Phil Borman about this and he replied:
'It's not really a problem with the trashcan, it's just that there are two ways in Qdos of saving a new version of a file... 1. Delete the old file, and save a new one with the same name 2. Overwrite the old file Method 1 uses DELETE, so the trashcan will save the old file for you. Method 2 doesn't use delete, so the trashcan doesn't get used. Some editors use method 1, some use method 2.'

This does not really explain why the real files on the hard disk get lost too. I would like reports from other users who have experienced the same problems.

It's QPC 2 You Too

Now that the final version of QPC 2 has leapt onto our screens there are a few challenging little questions that the programmers out there may wish to look into. Bear in mind here that I am just thinking aloud and passing on queries that other users have asked of me, I have no knowledge about the problems involved in these questions. Still, 'if you don't ask you don't get' as my old white haired mother says, so here goes.
First up are printers. I have one QPC 1 user who insists on trying to load a DOS printer driver when running the program and who then asks why the printer (a LEXMARK) does not print.

The answer to is that, although QPC 1 runs under DOS it is not a DOS program and therefore has no real contact with the underlying operating system. This, in effect means that nothing loaded in the AUTOEXEC.BAT or CONFIG.SYS files will have any meaning on the operation of the program unless there are specific call to it (ie the mouse driver and screen/disk operations etc.) Printing from the QDOS/SMSQ point of view is handled within the programs that generate the text and each has its own driver. ProWesS has made an admirable difference to this in having its own universal printer driver and maybe we should all be looking to producing new drivers for that and linking other applications into using. Rich Mellor's Q-Route also has a link which enables it to use other drivers on the system to print with (Text 87, Quill and Perfection) and Jochen Merz is now using a Universal driver in all of the new versions of his own programs (so far QD and QSpread) so we are getting there.

The question in my mind is how far does QPC 2 have contact with the surrounding Windoze environment? Three major questions have been asked here. The first, related to the above preamble, is: can it talk to the Windoze printer driver? This could be an interesting line of enquiry and could lead to an increase in the number of printers available to us. If someone could write a driver that would take the output of a program and pass it in the standard Windoze format to the driver within Windoze maybe this could do the trick. Not easy, I guess, but is it possible?

Next we have disk access. SMSQ/E has access to PC format floppies and its own QXL.WIN files on the hard disk/s of the PC but is it possible to gain access to the files on the other side of the fence? I realise that the standard devices displayed when you run QPC 2 or other file managers are hard coded into SMSQ/E itself but is there a way to install a device which would look at and access the other files? QXL owners were asking this a long time ago and, with the QXL it was a bit more difficult because of the way that it ran but now that QPC 2 is running within Windoze is this any easier? F. van der Planke has produced the QXL.WIN explorer which runs from Windoze and allows you to extract files from the QL side but I would like to be able to do the opposite as well - and to do it from SMSQ/E.

Finally we have the mouse question. Stephen Hall has been nagging away about this one because he has one of those neat scrolling mice and would like the scrolling action to work in QPC too. This is actually a subject which could be addressed in by superHermes as well. The extra keys, wheels and other things found on a lot of mice these days only produce a certain character at the serial port (try opening a port to screen and then activating the devices). SMSQ/E is a modular operating systems and it's modules can be added or removed quite easily. How about this: we could have a program which looks at the mouse port and asks you to activate one of the extra buttons or wheels. It then reads in the code and stores it. It would then ask you to allocate an action to that code (CURSOR UP/DOWN, PAGE UP/DOWN, RESIZE, MOVE WINDOW, etc). Once this has been set up it creates an SMSQ/E module and links it in. All of this may be total bilge of course so I rely on the more knowledgable of my readers to tell me why it cannot be done.

QL 2000

I realise that many of you will be suffering from the Millennium Bug - you are probably all sick of people talking about the upcoming event. I feel a similar sense of ennui but I have also been instrumental in trying to get QUANTA to organise a 'Year 2000' bash next year so that we could get together as many QDOS/SMSQ users from all over the world. Something like this needs a lot of organisation and the timing and location of the event are crucial to its success achieving this end.

John Taylor, QUANTA's treasurer and Robin Barker, QUANTA's chairman would like to hear from anyone who has a view on this - especially if you are travelling from another country or distant part of the British Isles. They would like nominations for both time and place for this event so they can get on and organise it. I have suggested that it be held at the Clevedon venue since that has a good choice of Hotels, some beautiful scenery and easy access to London, Wales, The Midlands and the West Country for those who want the event to be part of a larger visit to the UK. Send your suggestions to QUANTA at this address:

Robin Barker,
'Jelanda'
Wyndley Drive,
Sutton Coldfield B73 6EU
UK

I hope to see lots of you there.

I wonder if, when the date went from 999 to 1000 people were worrying about all the beads falling off their abacus?

QL Shows Oct./Nov. '99

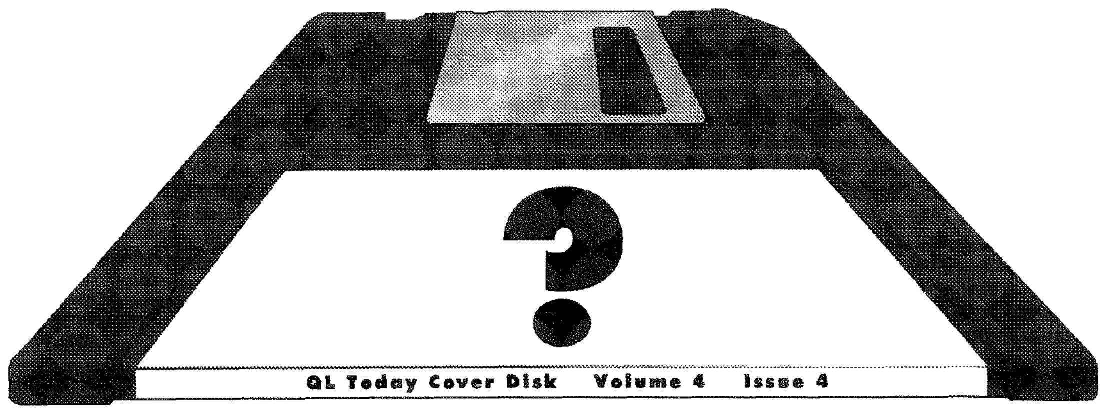
Only two shows have been reported to us for the next two months:

<u>Sat./Sun., 9./10. Oct. - Austria, Heidenreichstein.</u>
Not "just" a QL show, lots of other activities planned for the weekend (and what's next on the list of men's hobbies? Trains! You can see and ride real steam trains as well as model trains (here you probably "see" only). For more details see previous issues of QL Today or contact Jochen Merz Software.

<u>Sat., 20. November - The Netherlands, Eindhoven</u>
One of the "regular" meetings. Jochen Merz Software will be there. Starts at 10:00, ends at 16:30.

We hope to have more details about next years' shows in the next issue, especially about the Y2K-Show!

The Next Issue



A cover disk and (perhaps!) another little extra!
A review of MView, a QL file viewer!
The format of "PRINTER-DAT" explained!
Use Quill with HP Deskjet printers!
An assembler listing from George Gwilt (SET & ALTER works under SMSQ!
Plus the next parts of our regular series!

A plea - we need more articles - send us yours!